# Robustness of language recognition system to transmission channel

Raphaël Duroselle

UNIVERSITÉ DE LORRAINE

École doctorale IAEM Lorraine

# Robustesse au canal des systèmes de reconnaissance de la langue

## THÈSE

présentée et soutenue publiquement le jeudi 28 octobre 2021

pour l'obtention du

**Doctorat de l'Université de Lorraine**

**(mention informatique)**

par

Raphaël Duroselle

**Composition du jury**

| | | |
|---|---|---|
| *Président :* | M. Jean-Luc Gauvain | Directeur de Recherche, CNRS |
| *Rapporteurs :* | M. Jean-François Bonastre | Professeur, Université d'Avignon |
| | M. Sylvain Meignier | Professeur, Université du Mans |
| *Examinateur :* | M. Nicolas Courty | Professeur, Université de Bretagne Sud |
| *Encadrants :* | M. Denis Jouvet | Directeur de Recherche, INRIA |
| | M$^{me}$ Irina Illina | Maître de Conférence, Université de Lorraine |
| | M. Olivier Boëffard | Professeur, en détachement auprès du Ministère des Armées |

**Laboratoire Lorrain de Recherche en Informatique et ses Applications — UMR 7503**

Mis en page avec la classe thesul.

# Remerciements

Merci d'abord à mes directeur et co-directrice de thèse, Denis Jouvet et Irina Illina. Vous m'avez laissé très libre de l'orientation de ces travaux tout en veillant toujours à la rigueur des expériences et de la confrontation à la littérature. J'ai pu explorer largement les directions qui me tenaient à cœur en m'appuyant sur vos connaissances et votre expérience. Merci aussi d'avoir facilité mes aller-retour incessants entre Nancy et Paris.

Je remercie vivement Olivier Boëffard qui a construit ce sujet d'étude et a encadré ces travaux au Ministère des Armées. De nos discussions quotidiennes, je retiendrai la bienveillance et le goût de l'effort qui m'ont pendant plus de trois années poussé à progresser.

Messieurs Jean-François Bonastre et Sylvain Meigner, je vous remercie d'avoir accepté de rapporter cette thèse. Vos commentaires affûtés ont stimulé ma réflexion sur le positionnement de cette thèse dans la littérature. Merci également à Messieurs Jean-Luc Gauvain et Nicolas Courty. Vous avez par vos questionnements lors de la soutenance donné l'éclairage croisé nécessaire à l'analyse et au développement de ces travaux, à l'intersection du traitement de la parole et de l'adaptation de domaine.

Merci à Md Sahidullah de m'avoir guidé au début de la thèse, puis de s'être investi dans une collaboration au cours de la dernière année. C'est grâce à son expérience des compétitions internationales que nous avons conçu un système de reconnaissance de la langue performant pour la compétition *Oriental Language Recognition 2020*. Merci à Matthew Wiesner, John Steinberg et Kiran Karra de m'avoir fait découvrir les systèmes de reconnaissance de la parole et d'avoir exploré avec moi leur robustesse au domaine.

Je tiens à exprimer ma gratitude aux stagiaires, doctorants, ingénieurs, post-docs et permanents de l'équipe Multispeech pour l'atmosphère studieuse et propice aux échanges qui nous permet de mener notre recherche. Un merci appuyé à Imran Sheikh qui a pris le temps de relire la plupart de mes articles. Merci également à Christophe Cerisara d'avoir suivi cette thèse.

Je remercie le Ministère des Armées et la Direction Générale de l'Armement d'avoir financé et soutenu cette thèse, en particulier à travers la personne de Lauriane Aufrant qui m'a généreusement fait profiter de son expérience. Merci à mes collègues de m'avoir accueilli et stimulé.

Hélène Zganic au LORIA et Chantal Kerryjaouen à la DGA m'ont appuyé quotidiennement pour aplanir les difficultés administratives. Je les en remercie.

Sur une note plus personnelle, j'aimerais mentionner parmi tous ceux qui m'ont soutenu pendant trois ans Audrey, Eloïse, Héléna, Louise, Manon, Mathilde, Maylis, Alban, Daniel, Hugo, Jacques, Thibaut et Thomas. Merci à Lucile et Blanche. Merci Irène.

À l'orée de vingt ans d'instruction, primaire, secondaire, en classes préparatoires, école d'ingénieur et thèse, j'éprouve la plus grande reconnaissance pour mes professeurs. Enfin merci à mes parents.

# Sommaire

**Sommaire** **iii**

**Table des figures** **ix**

**Liste des tableaux** **xi**

**1 Introduction** **1**
1.1 Plan du document . . . . . 2
1.2 Résumé chronologique . . . . . 4

**2 Systèmes de reconnaissance de la langue** **9**
2.1 Tâche de reconnaissance de la langue . . . . . 9
2.1.1 Motivation . . . . . 9
2.1.2 Signal de parole . . . . . 10
2.1.3 Définition des classes . . . . . 10
2.1.4 Définition de la tâche . . . . . 10
2.1.5 Tâches voisines . . . . . 11
2.1.6 Notations . . . . . 12
2.1.7 Campagnes d'évaluation internationales . . . . . 12
2.2 Mesure de performance et évaluation . . . . . 14
2.2.1 Détection de la langue . . . . . 14
2.2.2 Évaluation des scores . . . . . 15
2.2.3 Évaluation de rapports de vraisemblance . . . . . 16
2.2.4 Évaluation d'un système de détection de plusieurs langues . . . . . 17
2.2.5 Point de vue sur les métriques . . . . . 20
2.3 Positionnement par rapport à la littérature . . . . . 20
2.3.1 Références . . . . . 20
2.3.2 Repères historiques . . . . . 21
2.4 Description des systèmes à l'état de l'art . . . . . 22
2.4.1 Architecture générale . . . . . 22
2.4.2 *Features* acoustiques . . . . . 25
2.4.3 Bottleneck features . . . . . 26
2.4.4 Réseau de neurones d'identification de la langue . . . . . 27

2.4.5 Classifieur final . . . 32
2.4.6 Calibration . . . 33
2.4.7 Fusion de systèmes . . . 33
2.4.8 Détection d'activité vocale . . . 33
2.4.9 Point de vue sur les architectures de système . . . 34
2.5 Directions actuelles de recherche . . . 34
2.5.1 Amélioration des recettes d'apprentissage . . . 34
2.5.2 Conditions difficiles . . . 35
2.5.3 Segmentation . . . 35
2.5.4 Positionnement de nos travaux . . . 35

**3 Le canal de transmission : un domaine pour les systèmes de reconnaissance de la langue 37**
3.1 La notion de domaine dans le cadre de l'apprentissage automatique . . . 38
3.1.1 Cadre de l'apprentissage automatique . . . 38
3.1.2 Apprentissage supervisé par minimisation du risque empirique . . . 39
3.1.3 Positionnement du travail . . . 41
3.2 Domaine défini par le canal de transmission . . . 42
3.2.1 Lien entre deux domaines . . . 42
3.2.2 Les facteurs de variabilité du signal de parole . . . 43
3.2.3 Le canal de transmission . . . 44
3.2.4 Nature de la notion de domaine . . . 44
3.2.5 Hypothèses associées au changement de canal . . . 45
3.3 Scénarios mettant en œuvre différents domaines . . . 46
3.3.1 Adaptation de domaine . . . 46
3.3.2 Apprentissage multi-domaines . . . 46
3.3.3 Généralisation à un nouveau domaine . . . 46
3.4 Méthodes de limitation de l'impact du domaine sur la performance des systèmes . . . 47
3.4.1 Réduction de l'écart entre les domaines . . . 47
3.4.2 Un traitement dépendant du domaine . . . 51
3.4.3 Autres approches . . . 52
3.4.4 Comparaison des approches . . . 52
3.4.5 Utilisation de données non annotées . . . 52
3.5 Analyse de systèmes . . . 53
3.6 Positionnement de ce travail . . . 54

**4 Robustesse d'un système de reconnaissance de la langue au canal de transmission 57**
4.1 Impact du changement de canal de transmission . . . 57
4.1.1 Test sur un canal inconnu . . . 58
4.1.2 Entraînement sur plusieurs canaux . . . 61
4.2 Régularisation de la fonction de coût du réseau d'identification de la langue 64
4.2.1 Principe . . . 64

4.2.2 Régularisation de la fonction de coût du réseau d'identification de la langue . . . 65
4.2.3 Fonctions mesurant l'écart entre les domaines . . . 66
4.2.4 Fonctions de coût de type *metric learning* . . . 67
4.2.5 Alignement entre données parallèles . . . 68
4.2.6 Démarche expérimentale . . . 69
4.3 Adaptation de domaine non supervisée . . . 69
4.3.1 Description du scénario . . . 69
4.3.2 Mise en œuvre de la régularisation . . . 69
4.3.3 Contexte expérimental . . . 70
4.3.4 Sélection de la couche régularisée . . . 71
4.3.5 Comparaison des fonctions de coût . . . 73
4.3.6 Adaptation simultanée vers plusieurs canaux . . . 74
4.3.7 Application au réseau d'identification de la langue d'un système *x-vector* . . . 75
4.3.8 Comparaison à d'autres méthodes d'adaptation de domaine . . . 76
4.3.9 Effet des hyper-paramètres . . . 78
4.3.10 Conclusion sur l'adaptation non supervisée . . . 80
4.4 Apprentissage multi-domaines . . . 81
4.4.1 Corpus utilisé . . . 81
4.4.2 Système utilisé . . . 82
4.4.3 Réduction de l'écart entre les domaines . . . 82
4.4.4 Regrouper les *x-vector* par langue . . . 84
4.4.5 Régularisation par *metric learning* . . . 84
4.4.6 Conclusion sur l'apprentissage multi-domaines . . . 85
4.5 Généralisation à un domaine inconnu . . . 86
4.5.1 Description du scénario . . . 86
4.5.2 Corpus utilisé . . . 86
4.5.3 Systèmes utilisés . . . 87
4.5.4 Stratégie adoptée . . . 87
4.5.5 Analyse des performances . . . 87
4.5.6 Conclusion sur la généralisation à un nouveau domaine . . . 88
4.6 Conclusion . . . 89

**5 Analyse des systèmes** **91**
5.1 Configurations de l'espace des représentations . . . 91
5.1.1 Visualisations de l'espace des représentations . . . 91
5.1.2 Configuration expérimentale . . . 92
5.1.3 Analyse des représentations . . . 95
5.1.4 Bilan des observations . . . 99
5.2 Caractérisation des représentations par la performance de classifieurs . . . 101
5.2.1 Notations . . . 101
5.2.2 Méthode . . . 102

5.2.3 Configuration expérimentale . . . 102
5.2.4 Performances . . . 102
5.2.5 Conclusion sur la performance des classifieurs . . . 103
5.3 Caractérisation directe des espaces de représentation . . . 104
5.3.1 Rapports de covariance . . . 104
5.3.2 Mesure de divergence . . . 105
5.4 Analyse des représentations de systèmes d'identification de la langue . . . 105
5.4.1 Motivation de l'analyse . . . 105
5.4.2 Configuration expérimentale . . . 106
5.4.3 Rapports de covariance . . . 106
5.4.4 Mesure de divergence . . . 108
5.4.5 Conclusion sur l'analyse d'un système de reconnaissance de la langue 112
5.5 Conclusion . . . 112

**6 Étude de l'extracteur de *bottleneck features* 115**
6.1 Positionnement par rapport à la littérature récente . . . 116
6.1.1 Différentes propriétés des *features* en fonction de l'architecture et de la recette d'entraînement . . . 116
6.1.2 Modèles dits *de bout en bout* . . . 117
6.1.3 Stratégies d'entraînement . . . 117
6.2 Modèle de reconnaissance de la parole de bout en bout multilingue . . . 119
6.2.1 Idée . . . 119
6.2.2 Description du modèle de reconnaissance de la parole multilingue . 119
6.2.3 Performance . . . 120
6.2.4 Conclusion sur les *bottleneck features* extraits d'un modèle de reconnaissance de la parole *de bout en bout* . . . 125
6.3 Étude de la stratégie d'entraînement . . . 127
6.3.1 Modèle conjoint d'identification de la langue et de reconnaissance de la parole . . . 127
6.3.2 Choix d'architecture . . . 128
6.3.3 Définition des stratégies . . . 128
6.3.4 Comparaison des stratégies . . . 130
6.3.5 Apprentissage multi-tâches . . . 133
6.4 Conclusion . . . 135

**7 Conclusion 137**

**Annexes 143**

**A Description des corpus 143**
A.1 *NIST Language Recognition Evaluations* . . . 143
A.1.1 Description des corpus d'origine . . . 143
A.1.2 Constitution des corpus utilisés dans nos expériences . . . 143
A.2 Robust Automatic Transcription of Speech (*RATS*) . . . 146

A.2.1 Description des corpus d'origine . . . 146
A.2.2 Constitution des corpus utilisés dans nos expériences . . . 148
A.3 *Oriental Language Recognition* (*OLR*) . . . 149
A.3.1 Description des corpus d'origine . . . 149
A.3.2 Constitution des corpus utilisés dans nos expériences . . . 149
A.4 Reconnaissance de la parole . . . 149
A.4.1 Babel . . . 151
A.4.2 Langue anglaise . . . 151
A.5 Corpus utiles aux augmentations de données . . . 153
A.5.1 *MUSAN : A Music, Speech, and Noise Corpus* . . . 153
A.5.2 *Room Impulse Response* (*RIR*) . . . 153

**B Résultats complémentaires sur le corpus *OpenSAD 15* 155**
B.1 Perte de performance sur des canaux inconnus du système *CNN* . . . 155
B.2 Adaptation de domaine non supervisée avec le système *BNF-CNN* . . . 156

**C Système de détection d'activité vocale 159**
C.1 Description du système . . . 159
C.2 Performance . . . 160

**D Participation à la compétition *Oriental Language Recognition 2020* 161**
D.1 Utilisation des données . . . 162
D.1.1 Corpus . . . 162
D.1.2 Augmentations de données . . . 163
D.2 *Features* à l'échelle de la trame . . . 163
D.2.1 Représentations spectrales . . . 163
D.2.2 *Bottleneck features* . . . 164
D.3 Entraînement des réseaux de neurones d'identification de la langue . . . 164
D.3.1 Recette d'entraînement . . . 164
D.3.2 Exploration d'autres fonctions de coût . . . 164
D.4 Entraînement du *GMM* . . . 165
D.5 Fusion et calibration . . . 165
D.6 Expériences . . . 165
D.6.1 Entraînement de l'extracteur de *bottleneck features* . . . 165
D.6.2 Sélection d'un modèle robuste . . . 165
D.6.3 Conception du système final . . . 166
D.7 Analyse du système soumis . . . 168
D.8 Conclusion . . . 168

**E Analyse des représentations de systèmes de reconnaissance de la parole 171**
E.1 La tâche de reconnaissance de la parole . . . 171
E.2 Motivation de l'analyse . . . 172
E.3 Condition expérimentale . . . 172

E.4 Méthode d'analyse . . . . . 173
E.5 Mesures de divergence par corpus . . . . . 175
E.6 Mesures de divergence par locuteur . . . . . 175
E.7 Comparaison de systèmes . . . . . 178
E.8 Conclusion sur l'analyse d'un système de reconnaissance de la parole . . . 179

**Glossaire** **181**

**Publications** **185**

**Bibliographie** **187**

**Résumé** **207**

**Abstract** **209**

# Table des figures


2.1 Architecture d'un système de reconnaissance de la langue . . . . . 23

4.1 Découpe du corpus *RATS* pour l'évaluation de la performance sur un canal inconnu . . . . . 64
4.2 Découpe des corpus *RATS* et *OpenSAD 15* pour les expériences d'adaptation de domaine non supervisée . . . . . 70
4.3 Schéma du système de reconnaissance de la langue . . . . . 77
4.4 Valeurs des fonctions de coût au cours de l'entraînement par régularisation avec la *MMD* . . . . . 79

5.1 Schéma d'un bloc résiduel utilisé dans le modèle *ResNet-1D* . . . . . 93
5.2 Projection *t-SNE* des *embeddings* du corpus *RATS* pour le modèle entraîné sur le canal téléphonique . . . . . 97
5.3 Projection *t-SNE* des *embeddings* du corpus *RATS* pour le modèle entraîné sur neuf canaux . . . . . 97
5.4 Projection *t-SNE* des *embeddings* du corpus *NIST LRE 2011* . . . . . 98
5.5 Projections *t-SNE* des *embeddings* du corpus *NIST LRE 2011* pour certains sous-ensembles de langues . . . . . 100
5.6 Analyse de la variabilité due au canal de transmission pour cinq modèles sur le corpus *NIST LRE 2011* . . . . . 110
5.7 Analyse de la variabilité due au genre pour cinq modèles sur le corpus *NIST LRE 2011* . . . . . 111

6.1 Représentation de l'extraction de *bottleneck features* pour une séquence de trames acoustiques. . . . . 118
6.2 Schéma du modèle conjoint d'identification de la langue et de reconnaissance de la parole. . . . . 127
6.3 Illustration des stratégies d'apprentissage pour l'entraînement de l'extracteur de *bottleneck features*, le module d'identification de la langue et le décodeur de reconnaissance de la parole . . . . . 129

D.1 Matrices de confusion du système soumis à l'évaluation *OLR 2020* . . . . 169

E.1 Projection *LDA* de l'espace d'*embeddings* du modèle entraîné sur *librispeech*.174

E.2 Divergences entre les *embeddings* correspondant au corpus de test et ceux du corpus de validation pour le modèle entraîné sur *librispeech*. . . . . 176
E.3 Divergences entre groupes d'*embeddings* pour le modèle entraîné sur le corpus *librispeech* pour différentes couches et mesures de divergence. . . . 177
E.4 Divergences entre groupes d'*embeddings* pour la couche de sortie du modèle *Conformer*. . . . . 178

# Liste des tableaux


4.1 Architecture du système *GMM*. . . . . 58
4.2 Architecture du système basé sur un réseau de neurones d'identification de la langue. . . . . 59
4.3 Performance sur les corpus de l'évaluation *Oriental Language Recognition 2020* . . . . . 60
4.4 Architecture du réseau *TDNN* . . . . . 62
4.5 Architecture du système *x-vector* . . . . . 63
4.6 Performance du système *x-vector* sur le corpus *RATS* . . . . . 65
4.7 Architecture du réseau de neurones convolutionnel . . . . . 71
4.8 Architecture du système *CNN*. . . . . 72
4.9 Performance du système *CNN* sur le corpus *OpenSAD 15* pour une adaptation entre les canaux G et D . . . . . 73
4.10 Comparaison des fonctions de régularisation avec le système *CNN* pour le corpus *OpenSAD 15* . . . . . 74
4.11 Adaptation simultanée vers plusieurs canaux du système *CNN* pour le corpus *OpenSAD 15* . . . . . 75
4.12 Performance de l'adaptation non supervisée du réseau d'identification de la langue dans le système *x-vector* sur le corpus *RATS* . . . . . 75
4.13 Performance sur le corpus *RATS* en fonction des méthodes d'entraînement du réseau *x-vector* et du classifieur final . . . . . 78
4.14 Architecture du système *x-vector bis*. . . . . 83
4.15 Performance du système *x-vector bis* sur les ensembles de test du corpus *NIST LRE 2011* . . . . . 84
4.16 Performance pour la tâche 1 de la compétition *OLR 2020* . . . . . 88

5.1 Architecture du système *ResNet-1D*. . . . . 94
5.2 Architecture du réseau *ResNet-1D* . . . . . 95
5.3 Performance des systèmes *ResNet-1D* sur le corpus *RATS* . . . . . 95
5.4 Performance de classifieurs sur le corpus *RATS* en fonction des méthodes d'entraînement de l'extracteur de représentations et du classifieur . . . . . 103
5.5 Analyse du système *CNN* entraîné sur le corpus *OpenSAD 15* avec plusieurs méthodes d'entraînement . . . . . 107

6.1 Architecture du système reposant sur un *TDNN* . . . . . 122
6.2 Comparaison de *features* sur les ensembles de test du corpus *OLR 2020* . 123
6.3 Performance de reconnaissance automatique de la parole en fonction de la recette d'entraînement du modèle de reconnaissance de la parole . . . . . 123
6.4 Comparaison de *bottleneck features* sur les ensembles de test du corpus *OLR 2020* . . . . . 124
6.5 Architecture du système de reconnaissance de la langue pour le corpus *NIST LRE 2007* . . . . . 126
6.6 Performance de reconnaissance de la langue sur le corpus *NIST LRE 2007* 127
6.7 Performance de reconnaissance de la langue de plusieurs stratégies d'entraînement et architectures de modèles sur le corpus *NIST LRE 2007* . . . 132
6.8 Performance d'un entraînement multi-tâches sur le corpus *NIST LRE 2007* 134

A.1 Caractéristiques des corpus *NIST LRE*. . . . . 144
A.2 Caractéristiques du corpus *Callfriend*. . . . . 145
A.3 Caractéristiques du corpus d'entraînement *NIST SRE 2008*. . . . . 145
A.4 Distribution des langues du corpus *NIST LRE 2011* en fonction de leur présence sur les deux canaux dans les ensembles d'entraînement et de test 147
A.5 Caractéristiques du corpus *RATS SAD* . . . . . 148
A.6 Caractéristiques du corpus *RATS KWS* . . . . . 148
A.7 Caractéristiques du corpus *OpenSAD 15* . . . . . 148
A.8 Caractéristiques des corpus *Oriental Language Recognition*. . . . . 150
A.9 Utilisation des corpus *OLR 2020* pour nos trois séries d'expériences. . . . 151
A.10 Caractéristiques du corpus *Babel* . . . . . 152

B.1 Performance du système *CNN* sur le corpus *OpenSAD 15* . . . . . 156
B.2 Architecture du système *BNF-CNN* . . . . . 157
B.3 Performance du système *BNF-CNN* sur le corpus *OpenSAD 15* . . . . . 158
B.4 Performance de différentes méthodes d'entraînement du réseau convolutionnel du système *BNF-CNN* pour le corpus *OpenSAD 15* . . . . . 158
B.5 Performance de différentes méthodes d'entraînement du réseau convolutionnel du système *BNF-CNN* pour le corpus *OpenSAD 15* . . . . . 158

C.1 Performance du système de détection d'activité vocale sur l'ensemble de test du corpus *RATS SAD* . . . . . 160

D.1 Performance de reconnaissance de la parole pour différentes recettes d'entraînement . . . . . 166
D.2 Performance de reconnaissance de la langue pour les expériences de *sélection d'un modèle robuste* sur le corpus *OLR 2020* . . . . . 166
D.3 Performance pour les expériences de *conception du système final* sur le corpus *OLR 2020*. . . . . 167

E.1 Architecture du système de reconnaissance de la parole de bout en bout. . 173

# 1

# Introduction

La tâche de reconnaissance de la langue consiste à prédire la langue utilisée dans un enregistrement audio contenant de la parole. L'étude d'un tel traitement trouve sa justification dans ses applications pratiques : l'orientation vers un opérateur parlant la bonne langue sur une plateforme téléphonique, un pré-traitement dans un système de reconnaissance de la parole multilingue ou bien des applications militaires. La tâche peut être réalisée par un système automatique. Depuis 1996 les évaluations organisées par le *National Institute of Standards and Technology* (*NIST*, États-Unis d'Amérique) permettent une comparaison objective des systèmes les plus performants dans les mêmes conditions expérimentales. L'évaluation la plus récente a eu lieu en 2017. Elle a marqué la domination d'un nouveau type de systèmes, dont le module principal est un réseau de neurones profond entraîné à identifier la langue pour l'ensemble du segment de signal considéré. C'est ce type de systèmes que nous utilisons dans notre étude.

La majorité de l'effort de recherche dans le domaine de la reconnaissance de la langue s'est concentré sur des données téléphoniques, en raison des applications principales des systèmes développés. Comme les systèmes reposent sur un apprentissage automatique, c'est-à-dire sur la sélection de leurs paramètres en fonction d'un corpus de données d'entraînement, ils sont exposés à une perte de performance lors d'un changement entre les caractéristiques des données d'entraînement et de test. Dans ce travail, nous nous intéressons à la variabilité due au canal de transmission. En particulier, nous étudions des enregistrements téléphoniques, des émissions de radio et des canaux de transmission radio. Le problème de la robustesse des systèmes au canal de transmission est abondamment étudié, au moins depuis l'évaluation du *NIST* de 2009. Des méthodes ont été proposées, dont certaines sont applicables aux systèmes basés sur un réseau de neurones d'identification de la langue, approche dominante depuis 2017. Ce problème n'est cependant pas résolu et les systèmes continuent à subir une perte de performance lorsqu'ils sont exposés à des canaux de transmission différents lors de l'entraînement et de l'évaluation.

Ce travail vise à répondre à des questions d'intérêt pragmatique. Comment obtenir la meilleure performance dans toutes les conditions lorsque le corpus d'entraînement comprend plusieurs canaux de transmission ? Comment concevoir un système susceptible d'obtenir de bonnes performances sur un nouveau canal, *a priori* inconnu ? Dans quelle

mesure l'emploi d'enregistrements non annotés du canal d'intérêt augmente-t-elle la robustesse du système ? Les réponses à ces questions permettront d'orienter au mieux les efforts d'un concepteur de systèmes de reconnaissance de la langue, vers une acquisition de données du canal visé, une annotation de données existantes ou le développement d'algorithmes spécifiques.

Notre thèse propose d'explorer certaines pistes d'étude offertes spécifiquement par les nouveaux systèmes reposant sur un réseau de neurones d'identification de la langue. Nos contributions peuvent être regroupées selon trois directions. D'abord, nous proposons une méthode d'augmentation de la robustesse du système au canal de transmission, reposant sur la régularisation de la fonction de coût utilisée lors de l'entraînement du réseau de neurones, et la déclinons pour trois scénarios différents : adaptation de domaine non supervisée, apprentissage multi-domaines et généralisation à un domaine inconnu. Ensuite, nous introduisons des outils d'analyse des systèmes du point de vue de la robustesse au canal de transmission. Enfin nous proposons une nouvelle version du premier module du système, l'extracteur de *bottleneck features*, qui, tout en améliorant la performance, permet d'intégrer l'entraînement de ce module avec le réseau d'identification de la langue, ouvrant la voie à une régularisation conjointe de ces deux modules.

## 1.1 Plan du document

Le chapitre 2 présente le cadre des systèmes de reconnaissance de la langue que nous étudions. Nous définissons la tâche de reconnaissance de la langue et ses modalités d'évaluation puis décrivons le type de systèmes sur lequel porte notre étude : les systèmes reposant sur un réseau principal d'identification de la langue. Nous tentons de dresser un panorama assez large des différentes variantes introduites dans les années 2016 à 2021, dans le but de concentrer notre étude sur les éléments partagés par les systèmes.

Nous formalisons ensuite dans le chapitre 3 le problème de robustesse au canal de transmission grâce à la notion de domaine. Nous précisons les hypothèses associées à un domaine défini par un canal de transmission, puis définissons plusieurs scénarios d'apprentissage où le domaine peut induire une perte de performance. En analysant les méthodes introduites dans ces scénarios, nous dressons un inventaire des approches utilisées pour limiter l'impact du canal de transmission sur la performance d'un système.

Une de ces approches est la construction de représentations du signal invariantes entre les domaines. De cette façon, le système se comportera de la même façon sur tous les domaines. Dans le chapitre 4, nous étudions une méthode visant à l'obtention de représentations invariantes par le réseau de neurones d'identification de la langue : la régularisation de sa fonction de coût. Nous analysons deux types de fonction pour réaliser cette régularisation : des fonctions mesurant l'écart entre les domaines et des fonctions de *metric learning*. Cette étude s'inscrit dans une dynamique assez large entre 2019 et 2021 où de nombreux travaux ont augmenté la robustesse de systèmes de reconnaissance de la langue ou de reconnaissance du locuteur à différents facteurs de variabilité en modifiant la fonction de coût du réseau de neurones central du système. Nous montrons comment cette méthode peut être mise en œuvre pour plusieurs scénarios d'apprentissage. Un scénario

est défini par les domaines disponibles lors de l'entraînement et sur lesquels l'évaluation est réalisée. Nous étudions trois scénarios.

Le premier est l'adaptation de domaine non supervisée. Nous développons un système devant fonctionner sur un canal de transmission radio en utilisant uniquement un corpus annoté de données téléphoniques et un corpus non annoté d'enregistrements radios. C'est un scénario de grand intérêt pratique puisque des corpus importants de données téléphoniques ont été constitués par la communauté académique d'une part, et qu'il est réaliste pour de nombreuses applications pratiques de collecter des données non annotées correspondant au contexte d'application d'autre part. Pour ce scénario, nous comparons plusieurs fonctions de coût mesurant l'invariance entre les domaines. Nous sélectionnons la *maximum mean discrepancy* et montrons que la régularisation du réseau avec cette fonction de coût est supérieure à l'adaptation d'autres modules du système.

L'apprentissage multi-domaines est un scénario où plusieurs domaines sont compris à la fois dans les corpus d'entraînement et d'évaluation. Il correspond au cadre des évaluations du *NIST* et à beaucoup d'applications pratiques où les concepteurs de systèmes cherchent à tirer parti du maximum de données disponibles et agglomèrent donc des corpus d'origines différentes. Nous avons montré l'intérêt des deux types de fonctions de coût pour ce scénario et nous en servons pour illustrer leurs propriétés respectives. Les fonctions de coût mesurant l'écart entre les domaines n'utilisent pas d'annotations en langue mais se servent d'annotations en domaine. À l'inverse, les fonctions de *metric learning* ne se servent pas d'annotations en domaine et peuvent donc réduire des variabilités inconnues, mais nécessitent une annotation en langue.

Le scénario le plus difficile est la généralisation à un domaine inconnu, non observé dans le corpus d'entraînement. Nous l'avons exploré à l'occasion de la compétition *Oriental Language Recognition 2020*, pendant asiatique des évaluations *NIST*, à laquelle nous avons participé. Pour ce scénario, les méthodes de régularisation permettaient de produire des modèles plus robustes au canal de transmission mais peu performants. Leur combinaison avec des modèles plus performants mais moins robustes a apporté un gain important de performance.

Au cours du développement de cette méthode de régularisation, nous avons introduit des outils pour mesurer quantitativement la variabilité des représentations produites par le système au canal de transmission. Nous les décrivons dans le chapitre 5. Ces outils sont le calcul de rapports entre les covariances inter-classes et intra-classes et la mesure de divergences dans des espaces de représentation. L'utilisation de tels outils est nouvelle en reconnaissance de la langue, la plupart des travaux se limitant souvent à des analyses reposant sur des visualisations à deux dimensions de l'espace des représentations.

Ces deux outils permettent de comprendre certaines propriétés des systèmes. D'abord les fonctions de régularisation visant à l'invariance produisent des systèmes aux caractéristiques différentes des systèmes entraînés sur un corpus multi-domaines. Les représentations du signal restent séparées entre les domaines pour des couches profondes du réseau lors d'un apprentissage multi-domaines alors que la régularisation impose un alignement des représentations sur ces couches. De plus, l'alignement entre les domaines dans un espace de représentation améliore la discriminabilité des représentations, et ce pour chaque

domaine. Nous rendons également compte quantitativement de la différence de comportement entre les fonctions de régularisation de mesure de l'écart entre les domaines et de *metric learning*.

Enfin le chapitre 6 introduit une nouvelle recette d'entraînement d'un module important du système de reconnaissance de la langue : l'extracteur de *bottleneck features*. Nous appliquons à la tâche de reconnaissance de la langue le développement récent de systèmes de reconnaissance de la parole reposant sur un réseau de neurones de bout en bout. Nous utilisons ce type de modèles pour produire des représentations multilingues du signal et obtenons une meilleure performance de reconnaissance de la langue avec une recette d'entraînement simplifiée en comparaison des approches classiques.

La comparaison de cet extracteur de *bottleneck features* reposant sur un réseau de reconnaissance de la parole de bout en bout avec un extracteur classique révèle que l'amélioration de la performance pour ce nouveau modèle provient majoritairement de l'architecture et non de la supervision par la tâche de reconnaissance de la parole. Ce constat nous permet de développer une stratégie d'entraînement conjoint des deux modules. Cela constitue une étape vers une potentielle intégration de tous les modules du système de reconnaissance de la langue en un unique modèle.

Nous reportons en annexe la description des corpus utilisés dans nos expériences (annexe A). Nous présentons également des expériences réalisées sur le corpus *OpenSAD 15* qui complètent le chapitre 4 (annexe B) et un des systèmes de détection d'activité vocale utilisés (annexe C). Les expériences réalisées dans le cadre de notre participation à la compétition *Oriental Language Recognition 2020* sont rapportées en annexe D. Dans l'annexe E, nous appliquons enfin à un système de reconnaissance de la parole une des méthodes d'analyse des représentations présentées dans le chapitre 5.

## 1.2 Résumé chronologique

Les expériences et les résultats rapportés dans cette thèse laissent entrevoir une diversité de corpus et d'architectures de systèmes. Nous avons ainsi réalisé des expériences avec les corpus *OpenSAD 15*, *RATS*, *NIST LRE 2011*, *OLR 2020* et *NIST LRE 2007*. Les systèmes de reconnaissance de la langue étudiés obéissent au même paradigme général : un module central constitué d'un réseau de neurones prend une séquence de *features* en entrée et est entraîné à prédire la langue. Dans notre travail, ce module a été réalisé alternativement par trois architectures : un réseau convolutionnel, l'architecture *time delay neural network* (*TDNN*) standard et un modèle résiduel. Ces modèles ont été utilisés, soit pour réaliser une classification de bout en bout, soit pour produire des représentations qui sont ensuite fournies à un classifieur final, et nous leur avons parfois adjoint un extracteur de *bottleneck features*. Enfin nous utilisons souvent des métriques d'évaluation faisant fi de la calibration des scores, et alternativement les métriques standards de la communauté académique qui nécessitent la production de rapports de vraisemblance calibrés. Nous espérons que cette diversité des configurations expérimentales dans lesquelles elles ont été validées permet d'asseoir les approches exposées lors de cette thèse. Si chaque série d'expériences a à cœur de figer les configurations expérimentales afin de mettre

en lumière le seul paramètre étudié lors de l'expérience, le lecteur avide de conclusions plus générales pourra être frustré de ne pas trouver dans ce document une comparaison exhaustive des systèmes étudiés sur les corpus utilisés. Si le but d'une telle comparaison est la définition d'un système dominant pour une reconnaissance de la langue robuste au canal de transmission, il pourra trouver en annexe D notre proposition en ce sens soumise à la compétition *Oriental Language Recognition 2020*, qui a obtenu la meilleure performance pour deux des trois tâches proposées. Nous ne pensons pas qu'une telle recette universelle existe aujourd'hui. Notre travail vise à dégager des méthodes générales qui doivent ensuite être adaptées finement à chaque configuration expérimentale pour obtenir la meilleure performance.

La cohérence de nos choix de corpus et de systèmes est donc bien plus à chercher du côté de l'évolution de nos thématiques de recherche et de contraintes pragmatiques (disponibilité des corpus par exemple), que d'une amélioration progressive vers un état de l'art hypothétique. Elle doit donc être comprise de façon chronologique et nous rapportons maintenant les orientations successives de cette thèse.

Nous avons débuté ce travail sous l'angle de l'adaptation de domaine non supervisée. Nous avons utilisé le corpus *RATS* en raison du nombre important de canaux de transmission qu'il comporte et parce que c'est un corpus parallèle (un même signal est transmis à travers différents canaux de transmission) qui autorise des expériences particulières. De plus les canaux radios étaient un sujet particulièrement ambitieux pour une adaptation de domaine non supervisée en raison des fortes distorsions qu'ils occasionnent. Nous ne disposions initialement que du corpus *OpenSAD 15* puis avons pu faire l'acquisition des corpus *RATS SAD* et *RATS KWS* qui nous ont permis d'ajouter une langue et deux canaux de transmission à nos expériences. C'est dans ce cadre que nous avons mis au point la régularisation d'un réseau de neurones avec une mesure de divergence, et nous avons utilisé la mesure de rapports de covariance dans l'espace des représentations dans ce but. Nous avons graduellement augmenté la complexité du système à mesure que la méthode fonctionnait pour nous approcher d'un système de référence. Après des premières expériences avec un unique réseau convolutionnel réalisant la tâche de bout en bout, nous avons ajouté un extracteur de *bottleneck features* puis un classifieur final. Nous étions particulièrement attachés à l'utilisation d'un classifieur final dans le but de montrer que la régularisation du réseau augmentait l'invariance de ses représentations internes et ne se limitait pas à un alignement des scores. Comme nous n'avions pas de proposition originale pour le classifieur final ou pour le module de calibration, nous nous limitions alors à des mesures de performance faisant abstraction de l'erreur de calibration. Les travaux sur l'adaptation de domaine non supervisée ont fait l'objet de deux publications [Duroselle *et al.*, 2020a, Duroselle *et al.*, 2020c].

Après avoir apporté une solution à l'adaptation de domaine non supervisée sur le corpus *RATS*, nous avons souhaité étudier comment la méthode de régularisation de la fonction de coût du réseau de neurones pouvait être appliquée dans le cadre de référence de la communauté de reconnaissance de la langue : les évaluations du *NIST*. Nous avons travaillé sur le corpus de la campagne 2011 pour la simple raison qu'il s'agissait de l'évaluation la plus récente pour laquelle il était possible en 2019 d'acquérir les données sans

avoir participé à la compétition. Nous avons appliqué les métriques de référence pour ce corpus qui imposent de tenir compte de la calibration et ont une spécificité : elles exigent la production de rapports de vraisemblance pour toutes les paires de langues cibles. Toujours armés de notre système de référence en trois modules, nous avons remplacé la machine à vecteurs supports par un modèle Gaussien dans le classifieur final car il permettait de produire des rapports de vraisemblance pour toutes les paires de langues de façon plus naturelle. Le problème posé par le canal de transmission dans ce cadre était l'apprentissage multi-domaines avec deux canaux : le téléphone et des conversations téléphoniques retransmises lors d'émissions de radio. Nous avons d'abord vérifié que la méthode de régularisation basée sur la *maximum mean discrepancy* et développée pour l'adaptation non supervisée était également utile dans ce cadre. Puis nous avons voulu limiter la variabilité des représentations en nous passant des annotations en canal et avons donc exploré le *metric learning* pour ce type de systèmes. Il était clair que les deux familles de fonctions de régularisation avaient des effets différents sur l'espace des représentations. Pour en rendre compte, nous avons introduit la mesure de divergence entre groupes de représentations dans cet espace. Ces travaux ont été publiés en 2020 [Duroselle *et al.*, 2020b].

Une ouverture vers une autre tâche nous a été offerte lors du *workshop SCALE 2020*. Pendant deux mois, nous avons pu découvrir les systèmes de reconnaissance de la parole de bout en bout qui ont amélioré de façon importante l'état de l'art de cette tâche entre les années 2015 et 2020. Nous avons étudié ces systèmes dans le cadre d'un apprentissage multi-domaines et tenté de mettre en œuvre les méthodes de régularisation développées pour la reconnaissance de la langue, sans succès manifeste. En revanche, à la suite de ce *workshop*, nous avons montré que la mesure de divergences dans un espace de représentation permettait de rendre compte de façon non supervisée de la performance de ces systèmes sur des domaines différents des domaines d'entraînement. Ce travail est une collaboration avec Matthew Wiesner, John Steinberg et Kiran Karra.

Pendant l'été et l'automne 2020, nous avons participé à la compétition *Oriental Language Recognition 2020*. Le système soumis est décrit dans la publication [Duroselle *et al.*, 2021a]. La première tâche de l'évaluation s'inscrivait parfaitement dans notre sujet d'étude : la généralisation à un canal de transmission inconnu. Elle s'est révélée une opportunité d'évaluer en aveugle les fonctions de régularisation étudiées. Dans nos systèmes de reconnaissance de la langue précédents, nous utilisions un modèle d'extraction de *bottleneck features* pré-entraîné par *BUT / Phonexia*. Il ne pouvait cependant être mobilisé lors de la compétition car il avait été entraîné sur le corpus *Babel* qui ne faisait pas partie des données autorisées pour le développement des systèmes. Nous avons donc été contraints de développer notre propre système d'extraction de *bottleneck features* et avons mis en œuvre les enseignements de l'été en développant un réseau de neurones de reconnaissance de la parole de bout en bout multilingue pour extraire des *bottleneck features*. Ce travail a été réalisé avec Md Sahidullah.

Convaincus de l'intérêt de ce nouveau paradigme d'extraction de *bottleneck features*, nous avons voulu le comparer à une approche classique et donc au modèle de *BUT / Phonexia* que nous considérons comme une référence. Nous avons donc entraîné notre

nouveau modèle sur le même corpus de reconnaissance de la parole, *Babel*. Nous souhaitions évaluer la capacité intrinsèque de reconnaissance de la langue des *features* ainsi produits en faisant abstraction des difficultés introduites dans les campagnes *NIST* récentes. Nous avons choisi le corpus de l'évaluation 2007 pour deux raisons. D'abord il ne contient qu'un seul canal de transmission, téléphonique. Ensuite, les quatorze langues cibles de ce corpus appartiennent à des familles de langues différentes et ce cadre d'évaluation rend mieux compte de la performance moyenne d'un système de reconnaissance de la langue que le corpus de l'évaluation 2011, déjà utilisé, pour lequel les erreurs sont dominées par quelques paires de langues voisines. À l'issue de cette comparaison expérimentale, nous avons compris qu'il était préférable d'intégrer l'entraînement du modèle d'extraction de *bottleneck features* avec le réseau d'identification de la langue. Cette analyse est présentée dans le papier [Duroselle *et al.*, 2021b].

La prochaine étape logique de ce travail, non présentée dans ce document, serait l'application des méthodes de régularisation étudiées au modèle d'extraction des *bottleneck features*. La régularisation pourrait être appliquée directement au modèle de reconnaissance de la parole, ou bien indirectement par l'intermédiaire du réseau d'identification de la langue dans le cas d'un entraînement conjoint des deux modules.

# 2

# Systèmes de reconnaissance de la langue

Ce chapitre expose le cadre dans lequel nous nous plaçons : les systèmes de reconnaissance de la langue reposant sur un réseau de neurones d'identification de la langue. Nous définissons la tâche de reconnaissance de la langue et précisons ses modalités d'évaluation. Après un bref positionnement historique, nous décrivons précisément les systèmes étudiés et évoquons les pistes de recherche principalement étudiées en 2021.

Nous nous limitons ici à exposer les modalités d'évaluation des systèmes de reconnaissance de la langue communément adoptées par la communauté académique, ainsi qu'à décrire dans un cadre unifié les systèmes dominants dans la littérature et sur lesquels porte notre étude sur l'augmentation de la robustesse au canal de transmission. Pour une introduction plus générale aux fondements de la tâche de reconnaissance de la langue, nous recommandons la lecture de la revue de littérature de Haizhou Li [Li *et al.*, 2013].

## 2.1 Tâche de reconnaissance de la langue

### 2.1.1 Motivation

La reconnaissance de la langue est une tâche du domaine de la reconnaissance de formes. Elle consiste à prédire la langue utilisée dans un enregistrement de signal audio, supposé contenir de la parole, avec un algorithme sans intervention humaine, en se basant uniquement sur des caractéristiques du signal.

Un tel traitement trouve son intérêt dans plusieurs contextes applicatifs. Il peut d'abord apporter une information pertinente en soi, par exemple dans les applications militaires de surveillance. Il peut également permettre de filtrer certains signaux, par exemple sur une plateforme téléphonique, dans le but d'aiguiller chaque appel vers un employé parlant la langue détectée. Enfin, la reconnaissance de la langue est fondamentale dans les systèmes de reconnaissance de la parole multilingues afin de sélectionner les bons alphabet et lexique de transcription.

### 2.1.2 Signal de parole

Dans cette thèse, nous nous concentrons sur la reconnaissance de la langue basée uniquement sur le signal audio. D'autres travaux exploitent la complémentarité du texte ou de l'image avec le son avec des systèmes dits multimodaux [Choi et Wang, 2021].

Un signal audio est un signal monodimensionnel, variant au cours du temps. Il est capté par un ou plusieurs microphones. Dans ce travail, nous nous limitons aux signaux à un seul canal (correspondant à un seul microphone). Pour pouvoir faire l'objet d'un traitement informatisé, le signal analogique est échantillonné selon une fréquence d'échantillonnage et discrétisé par un opérateur de quantification. On parle alors de signal numérique.

### 2.1.3 Définition des classes

La reconnaissance de la langue se heurte au problème de la définition des différentes classes du problème de classification. Le catalogue *glottolog* [Hammarström *et al.*, 2021] dénombre 8516 langues. Cependant des nuances existent et la frontière entre les notions de langue et de dialecte, et de dialecte et d'accent ne sont pas simplement tranchées. Ainsi, la notion la plus communément admise de langue repose sur le concept d'intelligibilité mutuelle : si deux locuteurs se comprennent alors ils parlent la même langue, éventuellement selon différents dialectes. D'autre part, la notion d'intelligibilité mutuelle elle-même dépend parfois du vocabulaire employé. Par exemple, deux locuteurs parlant hindi et ourdou se comprennent toujours, sauf quand ils parlent de certains thèmes aux vocabulaires spécifiques. À l'inverse, on peut facilement faire l'expérience de deux locuteurs tentant d'utiliser la même langue et ne se comprenant pas (deux chercheurs chinois et français parlant un *globish* approximatif lors d'une conférence d'informatique).

Nous prenons ici le parti de ne pas nous appesantir sur la définition linguistique d'une langue. Pour chacune de nos expériences, nous définissons les différentes classes utilisées en attribuant le nom langue à des réalités linguistiques pouvant recouvrir la famille de langue, la langue ou le dialecte. Ce point de vue est globalement adopté par la communauté de reconnaissance de la langue qui utilise les mêmes types de modèles pour traiter les différents raffinements de la notion de langue. Ainsi, lors de nos différentes expériences certaines langues pourront appartenir ou non à la même classe, comme les dialectes de l'arabe, ou de l'anglais parlé par des locuteurs de nationalités différentes, différentes langues chinoises (mandarin, cantonais, minnan, wu) ou l'hindi et l'ourdou qui sont parfois regroupés en hindoustani.

### 2.1.4 Définition de la tâche

Une fois l'ensemble des classes précisé, il convient de définir la nature de la tâche de classification. La tâche la plus générale est appelée segmentation de la langue (*language diarization* en anglais) [Lyu *et al.*, 2013]. Elle consiste à découper un signal en plusieurs segments homogènes en langue et à prédire la langue employée dans chaque segment. Deux problèmes s'entremêlent alors, celui de la reconnaissance des langues employées et

celui de la prédiction précise de frontières temporelles définissant les différents segments.

Afin de se focaliser sur la tâche de reconnaissance de la langue, une large partie de la recherche fait l'hypothèse qu'une seule langue est employée dans l'enregistrement. La tâche devient donc celle de la prédiction d'une seule langue pour un signal audio. C'est dans ce cadre que nous nous plaçons.

Enfin deux modalités d'utilisation sont possibles pour la reconnaissance de la langue : l'identification et la détection [Brümmer et Van Leeuwen, 2006]. Lors d'une tâche d'identification, le système doit prédire la langue utilisée dans un enregistrement parmi un ensemble de langues connu par le système. Pour la détection ou vérification, étant donné un enregistrement et une langue cible, le système doit déterminer si l'enregistrement contient la langue cible. Les métriques standards d'évaluation des systèmes de reconnaissance de la langue se placent dans un cadre de détection. Cependant la plupart des systèmes actuels sont entraînés avec une tâche d'identification.

Pour les trois tâches de segmentation, d'identification et de détection, deux scénarios d'utilisation sont à envisager. Pour une évaluation fermée, les langues possibles de l'enregistrement présenté aux systèmes sont *a priori* connues. On ne fait pas cette hypothèse pour une évaluation ouverte, où un enregistrement de n'importe quelle langue peut être présenté au système. Ce scénario est plus difficile puisque le système doit être capable de rejeter une large classe de signaux sur lesquels nous ne faisons pas d'hypothèse.

### 2.1.5 Tâches voisines

D'autres informations peuvent être prédites à partir d'un enregistrement audio. Deux tâches d'importance doivent être évoquées afin de comprendre l'évolution du domaine de la reconnaissance de la langue : la reconnaissance du locuteur et la reconnaissance de la parole.

La reconnaissance du locuteur [Greenberg *et al.*, 2020] est très similaire à la tâche de reconnaissance de la langue puisqu'elle consiste également à prédire une étiquette pour un signal de durée variable. Pour cette tâche, on retrouve la problématique de segmentation (*diarization*), et de vérification qui consiste à déterminer si deux enregistrements ont été prononcés par le même locuteur. De plus, la recherche pour cette tâche est très dynamique en raison du champ d'application plus large de la reconnaissance du locuteur : personnalisation au locuteur pour les enceintes connectées, identification biométrique, preuve dans une procédure judiciaire, ainsi que les applications de surveillance. De nombreux systèmes de reconnaissance de la langue sont donc des transpositions directes de méthodologies de reconnaissance du locuteur.

La reconnaissance de la parole [Yu et Deng, 2015] est une tâche à la structure plus complexe : elle consiste à fournir une transcription en mots d'un enregistrement audio. C'est une tâche dite de séquence à séquence, par opposition aux reconnaissances de la langue ou du locuteur qui prédisent une étiquette unique. Cependant il est clair qu'un système de reconnaissance de la parole performant doit capturer des informations utiles à la reconnaissance de la langue : si une transcription pertinente peut être trouvée dans une certaine langue alors c'est vraisemblablement la langue employée dans l'enregistrement.

Nous verrons donc que les systèmes de reconnaissance de la langue peuvent faire intervenir des systèmes de reconnaissance de la parole.

### 2.1.6 Notations

Un système de reconnaissance de la langue $S$ est un système de reconnaissance de formes. Il opère sur un espace $\mathcal{X}$, représentant les signaux audios. Notons $\mathcal{L}$ l'ensemble des langues. Le système doit prédire des étiquettes $\mathcal{Y}$, classes correspondant à une information de reconnaissance de la langue.

$$S : \mathcal{X} \longrightarrow \mathcal{Y}$$

La définition de l'ensemble des classes varie en fonction de la tâche :

— identification parmi un ensemble fermé de $n$ langues $L_1, \ldots, L_n \in \mathcal{L}$

$$\mathcal{Y} = \{L_1, \ldots, L_n\}$$

— identification des $n$ langues $L_1, \ldots, L_n$ dans un cadre ouvert

$$\mathcal{Y} = \{L_1, \ldots, L_n, \neg\{L_1, \ldots, L_n\}\}$$

où $\neg\{L_1, \ldots, L_n\}$ est la classe correspondant à toutes les langues autres que les $n$ langues $L_1, \ldots, L_n$

— détection de la langue $L$

$$\mathcal{Y} = \{L, \neg L\}$$

où $\neg L$ est la classe correspondant au rejet de la langue $L$

L'ensemble des classes $\mathcal{Y}$ est parfois appelé l'ensemble des hypothèses.

Souvent, il est souhaitable que le système ne réalise pas directement une prédiction mais produise des scores pour chacune des hypothèses. Le score d'une classe est un réel représentant le degré de certitude du système en l'hypothèse que le signal testé appartienne à cette classe. De cette façon le système $S$ peut être décomposé en un extracteur de scores $s$ et une fonction de prédiction $p$.

$$S = p \circ s$$

où

$$s : \mathcal{X} \longrightarrow \mathbb{R}^{\mathcal{Y}}$$

$$p : \mathbb{R}^{\mathcal{Y}} \longrightarrow \mathcal{Y}$$

### 2.1.7 Campagnes d'évaluation internationales

Les travaux de la communauté académique de reconnaissance de la langue ont été façonnés par les campagnes d'évaluation internationales, organisées à partir de 1996. Celles-ci ont été initiées et sont toujours suivies par le *National Institute of Standards and Technology* (*NIST*, administration des États-Unis d'Amérique). Sur ce modèle, la

compétition *Oriental Language Recognition* a été organisée dans le cadre de l'*APSIPA* (*Asia Pacific Signal and Information Processing Association*) à partir de 2016.

Ces campagnes sont d'une grande importance pour la communauté académique. D'abord la constitution des corpus de données nécessaires à l'entraînement et à l'évaluation des systèmes est réalisée par l'organisateur, ce qui soulage les laboratoires d'une charge de travail importante. De plus, lors de certaines évaluations, un système de base est fourni par les organisateurs, ce qui réduit le coût d'entrée dans un domaine de recherche. Le cadre général d'évaluation des systèmes et les métriques d'évaluation utilisées par la communauté ont été définis par les évaluations du *NIST*. Enfin, ces campagnes d'évaluation permettent de comparer objectivement les méthodologies proposées avec une grande rigueur puisque les évaluations sont réalisées par les organisateurs.

Nous plaçons nos travaux dans ce cadre. Ainsi nous présentons des études réalisées sur les corpus des évaluations *NIST LRE* 2007 et 2011. Nous avons également participé au cours de cette thèse à l'évaluation *Oriental Language Recognition 2020*. Les tâches proposées lors des campagnes successives rendent compte de l'évolution des thématiques de recherche de la communauté académique.

#### *NIST Language Recognition Evaluations*

Les évaluations du *NIST* appelées *NIST Language Recognition Evaluations* (*NIST LRE*) ont été organisées en 1996, 2003, 2005, 2007, 2009, 2011, 2015 et 2017 [Sadjadi *et al.*, 2018b]. Les évaluations 1996 à 2007 concernaient uniquement des enregistrements téléphoniques. Douze langues cibles étaient utilisées en 1996 et 2003 : allemand, anglais (États-Unis), arabe (égyptien), coréen, espagnol (Amérique Latine), farsi, français (canadien), hindi, japonais, mandarin, tamoul et vietnamien. La campagne 2005 se concentrait sur neuf de ces langues. En 2007, en plus de la reconnaissance parmi quatorze langues assez distinctes, la reconnaissance d'une langue parmi un ensemble de langues de la même famille était évaluée. À partir de 2009, un autre canal de transmission a été introduit. Il est appelé *telephone bandwidth broadcast radio speech* ou *broadcast narrowband speech* (*BNBS*) et correspond à des appels téléphoniques retransmis lors d'émissions de radio. En 2009, l'évaluation était réalisée pour vingt-trois langues et les métriques étaient calculées pour certaines paires spécifiques. En 2011, vingt-quatre langues étaient utilisées et les systèmes étaient évalués pour les vingt-quatre paires de langues pour lesquelles ils commettaient le plus d'erreurs. À partir de 2015, l'évaluation s'est concentrée sur la reconnaissance d'une langue parmi un ensemble de langues de la même famille. Six familles étaient utilisées en 2015 : anglais, arabe, chinois, français, ibérique et slave. Le français a été abandonné en 2017. En 2017, un troisième canal de transmission correspondant à des vidéos extraites d'*Internet* a été introduit.

Ces évaluations utilisent plusieurs ensembles de test avec des durées de parole différentes : trois, dix et trente secondes. Nous les appelons dans ce document durées nominales car elles correspondent à une durée approximative, la détection d'activité vocale devant être réalisée par un système automatique.

***Oriental Language Recognition challenges***

Les évaluations *Oriental Language Recognition* (*OLR*) ont été organisées chaque année entre 2016 et 2020 par les institutions *Xiamen University*, *Tsinghua University*, *Duke-Kunshan University*, *Northwestern Polytechnical University* et *Speechocean*. Elles se concentrent sur des langues dites orientales et sur des enregistrements téléphoniques. En 2016 [Wang *et al.*, 2016], sept langues étaient utilisées : cantonais, coréen, mandarin, indonésien, japonais, russe et vietnamien. Trois autres ont été ajoutées en 2017 [Tang *et al.*, 2017a], kazakh, ouïghour et tibétain, ainsi que trois durées d'évaluation différentes, une, trois, ainsi que supérieure à trois secondes. En 2018 [Zhiyuan Tang, 2018], trois tâches étaient proposées : reconnaissance de la langue pour des segments courts, pour des langues proches (cantonais, coréen et mandarin) et parmi un ensemble ouvert. En 2019 [Tang *et al.*, 2019], les segments courts étaient maintenus et une tâche de reconnaissance de la langue pour un canal de transmission inconnu était introduite. Une tâche appelée *zero resource* était également proposée, elle visait à reconnaître une langue alors même qu'on ne disposait pas de données d'entraînement pour cette langue. En 2020 [Li *et al.*, 2020a], trois tâches étaient proposées : reconnaissance sur un canal inconnu, reconnaissance de dialecte et reconnaissance en environnement bruité. Nous avons participé et obtenu la première place aux première et troisième tâches de l'édition 2020.

## 2.2 Mesure de performance et évaluation

Dans cette partie, nous définissons les métriques utilisées pour caractériser la performance d'un système de reconnaissance de la langue dans ce travail. Elles ont été imposées à l'ensemble de la communauté à travers les campagnes d'évaluation *NIST LRE*. Ces métriques sont calculées sur un ensemble de test annoté $\mathcal{E}_{Te}$ de cardinal $N_{Te}$ et correspondant à un ensemble d'enregistrements $x \in \mathcal{X}$ et des étiquettes de langue $l \in \mathcal{L}$ associées.

$$\mathcal{E}_{Te} = \{(x_i,l_i)\}_{i=1}^{N_{Te}} \in (\mathcal{X} \times \mathcal{L})^{N_{Te}}$$

Les systèmes sont évalués pour une tâche de détection de la langue. Nous définissons les métriques d'évaluation d'un système de détection d'une langue, puis précisons comment évaluer des scores et non pas des décisions. Enfin nous expliquons comment agréger ces métriques pour plusieurs langues.

### 2.2.1 Détection de la langue

Pour une tâche de détection, nous nous intéressons uniquement à une langue cible, notée $L$. Le système peut donc réaliser deux prédictions : détecter la langue cible $L$ ou ne pas la détecter, nous noterons $\neg L$ cette seconde hypothèse. Par conséquent, pour un échantillon de test, quatre configurations sont possibles : vrai positif ($VP$), faux négatif ($FN$), faux positif ($FP$) et vrai négatif ($VN$). Deux types d'erreurs sont possibles : les fausses alarmes et les faux rejets. Les fréquences de ces erreurs sont notées $P_{FA}$ et $P_{FR}$.

$$P_{FA} = \frac{FP}{VN + FP} \quad P_{FR} = \frac{FN}{VP + FN}$$

La performance du système peut alors être évaluée en pondérant les deux types d'erreurs. Cette pondération dépend de l'application visée. Si le coût des fausses alarmes est fort, par exemple pour un système biométrique, on attendra un système très sûr de sa décision d'acceptation, tandis qu'on voudra minimiser la proportion de faux rejets pour un système chargé d'orienter vers un traducteur des enregistrements. On parle alors de point de fonctionnement. Un point de fonctionnement est caractérisé par trois paramètres. $P_{cible}$ est la probabilité *a priori* qu'un échantillon corresponde à la langue cible. $C_{FA}$ est le coût d'une fausse alarme et $C_{FR}$ le coût d'un faux rejet. Le coût estimé du système pour ce point de fonctionnement est obtenu à partir de la *detection cost function* (*DCF*).

$$DCF = P_{cible} P_{FR} C_{FR} + (1 - P_{cible}) P_{FA} C_{FA}$$

Lors des évaluations *NIST LRE* jusqu'à 2015 et pour les évaluations *OLR*, le point de fonctionnement choisi était ($P_{cible} = 0.5$,$C_{FR} = C_{FA} = 1$). C'est celui que nous utilisons dans toutes nos expériences. L'évaluation *NIST LRE 2017* a introduit un deuxième point de fonctionnement, en plus du premier, correspondant aux valeurs ($P_{cible} = 0.1$,$C_{FR} = C_{FA} = 1$).

### 2.2.2 Évaluation des scores

Afin d'être capable de modifier les propriétés du système en fonction du cadre d'utilisation, la communauté de reconnaissance de la langue a adopté des méthodologies d'évaluation se focalisant sur l'extracteur de scores $s$ en imposant une forme standard au module de prédiction $p$. De cette façon, en fonction du cadre d'application, la décision prise par le système peut être modifiée en changeant les paramètres de $p$ mais en conservant le même extracteur de scores.

La forme imposée à $p$ est la simple comparaison à un seuil $\sigma$. La langue $L$ est détectée si le score est supérieur à $\sigma$.

En faisant varier le seuil $\sigma$, on change les prédictions réalisées par le système pour certains échantillons et donc la distribution des erreurs. Une augmentation du seuil diminue le taux de fausse acceptation et augmente le taux de faux rejet. On peut donc définir des taux d'erreur en fonction du seuil : $P_{FR}^{\sigma}$ et $P_{FA}^{\sigma}$.

#### Points de fonctionnement servant à l'évaluation

Pour un point de fonctionnement caractérisé par les paramètres ($P_{cible}$,$C_{FR}$,$C_{FA}$), on peut sélectionner un seuil optimal qui permet de minimiser la valeur du *DCF* du système. Le coût associé à ce seuil est appelé le coût de détection minimal et noté *minDCF*.

$$minDCF = \min_{\sigma \in \mathbb{R}} [P_{cible} C_{FR} P_{FR}^{\sigma} + (1 - P_{cible}) C_{FA} P_{FA}^{\sigma}]$$

Un point de fonctionnement particulier est souvent utilisé afin de caractériser un système. Il correspond au seuil pour lesquels les taux de faux rejet et de fausse alarme sont égaux. Il est appelé le taux d'égale erreur et noté $EER$ (*equal error rate* en anglais).

$$EER \underset{P^{\sigma}_{FR}=P^{\sigma}_{FA}}{=} P^{\sigma}_{FR}$$

Les deux métriques $minDCF$ et $EER$ rendent compte de la capacité de discrimination des scores, c'est-à-dire de la performance d'un système pour lequel le seuil aurait été fixé de façon optimale en fonction de la distribution des scores. Elles ne prennent pas en compte l'erreur de calibration, qui est la part de l'erreur du système qui vient du choix du seuil [Brümmer et Van Leeuwen, 2006].

### 2.2.3 Évaluation de rapports de vraisemblance

Nous avons expliqué comment évaluer des scores et non directement des décisions, dans le but d'utiliser un même système dans différents cadres d'utilisation en faisant varier le seuil $\sigma$. Dès lors, comment fixer le seuil $\sigma$ en fonction du cadre d'utilisation ? Nous définirons ici le cadre d'utilisation par un point de fonctionnement ($P_{cible}$,$C_{FR}$,$C_{FA}$).

Une solution consiste à donner une interprétation probabiliste aux scores produits par le système. De cette façon, le seuil optimal peut être calculé directement à partir des paramètres du point de fonctionnement. Dans la communauté de reconnaissance de la langue et de reconnaissance du locuteur, les scores sont formulés en termes de rapports de vraisemblance. Dans la pratique, on utilise le logarithme des rapports de vraisemblance, noté $LLR$ (*log-likelihood ratio*). Formellement le logarithme du rapport de vraisemblance pour la détection de la langue $L$ admet la définition suivante.

$$LLR(x) = \ln \frac{p(x|L)}{p(x|\neg L)}$$

En faisant, l'hypothèse que les scores produits par le système correspondent à des rapports de vraisemblance, alors le seuil optimal à appliquer au point de fonctionnement ($P_{cible}$,$C_{FR}$,$C_{FA}$) est noté $\beta$ et admet l'expression suivante.

$$\beta = \frac{C_{FA} \times (1 - P_{cible})}{C_{FR} \times P_{cible}}$$

Lors des campagnes d'évaluation *NIST* et *Oriental Language Recognition*, ce point de vue est adopté. Les participants à l'évaluation doivent soumettre les scores prédits par leurs systèmes sous la forme de logarithmes de rapports de vraisemblance. L'opération de seuillage est ensuite appliquée par les évaluateurs avec la valeur théorique du seuil $\sigma = \ln(\beta)$, éventuellement pour plusieurs points de fonctionnement. La métrique utilisée est souvent le coût de détection $DCF^{\star}$.

$$DCF^{\star} = P_{cible} C_{FR} P^{\ln(\beta)}_{FR} + (1 - P_{cible}) C_{FA} P^{\ln(\beta)}_{FA}$$

### 2.2.4 Évaluation d'un système de détection de plusieurs langues

Les systèmes utilisés en pratique sont des systèmes de détection de plusieurs langues. Afin de rendre compte de la performance globale d'un tel système, il convient de proposer des métriques synthétisant la performance pour l'ensemble des langues visées. Trois types de métriques existent : celles qui agrègent les scores pour les différentes langues cibles, celles qui moyennent des métriques calculées par paire de langues et celles qui moyennent des métriques calculées par langue cible.

#### Définitions pour un système de détection de plusieurs langues

Un système de détection de plusieurs langues doit détecter un ensemble de langues cibles, noté $\mathcal{L}_\mathcal{T}$. Pour chaque langue cible $L_\mathcal{T}$, le système doit la distinguer d'un ensemble de langues dites non cibles, noté $\mathcal{L}_\mathcal{N}(L_\mathcal{T})$. Les langues non cibles pour chaque langue cible peuvent être connues lors du développement du système, on parle alors de détection de la langue parmi un ensemble fermé. Dans le cas contraire on parle d'évaluation ouverte. C'est un cadre d'évaluation plus difficile puisque le système doit être capable de rejeter des langues non vues lors de son entraînement.

Pour un système de détection de $n = |\mathcal{L}_\mathcal{T}|$ langues cibles, il est courant que l'ensemble des langues non cibles pour chaque langue cible soit l'ensemble des autres langues cibles. C'était le cas lors des évaluations *NIST* 2003 à 2009. On a alors :

$$\forall L_\mathcal{T} \in \mathcal{L}_\mathcal{T}, \quad \mathcal{L}_\mathcal{N}(L_\mathcal{T}) = \mathcal{L}_\mathcal{T} \backslash \{L_\mathcal{T}\}$$

Afin de concentrer l'évaluation sur une tâche plus difficile, il arrive également que l'on définisse un ensemble de langues non cibles en fonction de la proximité linguistique des langues. Ainsi, lors des évaluations *NIST* 2015 et 2017, les langues cibles étaient regroupées par familles de langues. Pour chaque langue cible, les langues non cibles étaient les autres langues de la même famille.

Pour chaque paire de langues cible $L_\mathcal{T}$ et non cible $L_\mathcal{N}$, on définit les vrais positifs $VP(L_\mathcal{T})$, faux négatifs $FN(L_\mathcal{T})$, faux positifs $FP(L_\mathcal{T},L_\mathcal{N})$ et vrais négatifs $VN(L_\mathcal{T},L_\mathcal{N})$.

$$VP(L_\mathcal{T}) = \sum_{i=1}^{N_{Te}} \mathbb{1}_{S_{L_\mathcal{T}}(x_i)=L_\mathcal{T}} \mathbb{1}_{l_i=L_\mathcal{T}} \quad FN(L_\mathcal{T}) = \sum_{i=1}^{N_{Te}} \mathbb{1}_{S_{L_\mathcal{T}}(x_i)\neq L_\mathcal{T}} \mathbb{1}_{l_i=L_\mathcal{T}}$$

$$VN(L_\mathcal{T},L_\mathcal{N}) = \sum_{i=1}^{N_{Te}} \mathbb{1}_{S_{L_\mathcal{T}}(x_i)\neq L_\mathcal{T}} \mathbb{1}_{l_i=L_\mathcal{N}} \quad FP(L_\mathcal{T},L_\mathcal{N}) = \sum_{i=1}^{N_{Te}} \mathbb{1}_{S_{L_\mathcal{T}}(x_i)=L_\mathcal{T}} \mathbb{1}_{l_i=L_\mathcal{N}}$$

#### Métriques agrégeant les scores

Les métriques de détection présentées dans les sections précédentes s'appliquent à la détection d'une seule classe. Pour se placer dans le même cadre, il est possible de regrouper les prédictions en quatre catégories en sommant sur les langues : vrais positifs, faux positifs, faux négatifs et vrais négatifs.

$$VP = \sum_{L_\mathcal{T} \in \mathcal{L}_\mathcal{T}} VP(L_\mathcal{T}) \quad FN = \sum_{L_\mathcal{T} \in \mathcal{L}_\mathcal{T}} FN(L_\mathcal{T})$$

$$FP = \sum_{L_\mathcal{T} \in \mathcal{L}_\mathcal{T}} \sum_{L_\mathcal{N} \in \mathcal{L}_\mathcal{N}(L_\mathcal{T})} FP(L_\mathcal{T},L_\mathcal{N}) \quad VN = \sum_{L_\mathcal{T} \in \mathcal{L}_\mathcal{T}} \sum_{L_\mathcal{N} \in \mathcal{L}_\mathcal{N}(L_\mathcal{T})} VN(L_\mathcal{T},L_\mathcal{N})$$

À partir de ces valeurs agrégées, il est possible de définir des taux de faux rejet et de fausse acceptation pour l'ensemble du système.

$$P_{FR} = \frac{FN}{VP + FN} \quad P_{FA} = \frac{FP}{VN + FP}$$

Ces métriques dépendent du seuil $\sigma$ qui est supposé identique pour tous les modules de prédiction $p_{L_\mathcal{T}}$. Par conséquent, en faisant varier $\sigma$, on peut s'intéresser à différents points de fonctionnement de système. On utilise couramment le taux d'égale erreur global du système, noté $EER$.

$$EER \underset{P^\sigma_{FR}=P^\sigma_{FA}}{=} P^\sigma_{FR}$$

**Moyenne par paire de langues**

Au lieu d'agréger les scores pour l'ensemble des paires de langues, le système peut être directement évalué par paire de langues puis ces métriques moyennées.

Considérons d'abord une paire de langues cible $L_\mathcal{T} \in \mathcal{L}_\mathcal{T}$ et non cible $L_\mathcal{N} \in \mathcal{L}_\mathcal{N}(L_\mathcal{T})$. On peut définir les taux de faux rejet et de fausse acceptation pour cette paire de langues.

$$P_{FR}(L_\mathcal{T}) = \frac{FN(L_\mathcal{T})}{VP(L_\mathcal{T}) + FN(L_\mathcal{T})} \quad P_{FA}(L_\mathcal{T},L_\mathcal{N}) = \frac{FP(L_\mathcal{T},L_\mathcal{N})}{VN(L_\mathcal{T},L_\mathcal{N}) + FP(L_\mathcal{T},L_\mathcal{N})}$$

À partir de ces valeurs, on peut définir un coût de détection $DCF(L_\mathcal{T},L_\mathcal{N})$ pour cette même paire de langues.

$$DCF(L_\mathcal{T},L_\mathcal{N}) = P_{cible} C_{FR} P_{FR}(L_\mathcal{T}) + (1 - P_{cible}) C_{FA} P_{FA}(L_\mathcal{T},L_\mathcal{N})$$

Ces métriques sont ensuite moyennées sur plusieurs paires de langues afin de caractériser l'ensemble du système. La métrique la plus largement utilisée par le *NIST* et les campagnes *OLR* est la moyenne sur toutes les paires de langues cibles et non cibles. Elle est appelée $C_{avg}$ pour coût moyen.

$$C_{avg} = \frac{1}{|\mathcal{L}_\mathcal{T}|} \sum_{L_\mathcal{T} \in \mathcal{L}_\mathcal{T}} \left[ P_{cible} C_{FR} P_{FR}(L_\mathcal{T}) + \frac{(1 - P_{cible})}{|\mathcal{L}_\mathcal{N}(L_\mathcal{T})|} \sum_{L_\mathcal{N} \in \mathcal{L}_\mathcal{N}(L_\mathcal{T})} C_{FA} P_{FA}(L_\mathcal{T},L_\mathcal{N}) \right]$$

Il a également été proposé de moyenner les coûts de détection pour certaines paires de langues particulières. Nous utilisons alors la notation $APD$ pour *Average Pair Detection*

*cost* (moyenne sur les paires du coût de détection). Notons $\mathcal{L}_P$ l'ensemble des paires de langues utilisées.

$$APD = \frac{1}{|\mathcal{L}_P|} \sum_{(L_\mathcal{T},L_\mathcal{N})\in\mathcal{L}_P} DCF(L_\mathcal{T},L_\mathcal{N})$$

Lors de l'évaluation *NIST LRE 2011*, la métrique $APD$ était utilisée. Le système était évalué pour une tâche de détection de vingt-quatre langues. L'ensemble $\mathcal{L}_P$ était constitué de vingt-quatre paires de langues parmi les $\frac{24\times23}{2} = 276$ paires possibles. Ces paires étaient celles qui donnaient le plus grand $DCF$ pour l'ensemble de test constitué de segments de durée nominale trente secondes. Lors de cette évaluation, la métrique $APD$ permettait donc de s'intéresser aux paires de langues pour lesquelles le système réalisait le plus d'erreurs.

Toutes les métriques présentées dans ce paragraphe dépendent du seuil $\sigma$ des modules de détection, qui est supposé identique pour toutes les langues cibles. On peut donc noter $C^\sigma_{avg}$ et $APD^\sigma$.

Pour le calcul des métriques $C_{avg}$ et $APD$, on applique souvent implicitement le seuil $\beta$ défini par le point de fonctionnement aux scores produits par le système, qui sont considérés comme des rapports de vraisemblance. Faire varier le seuil, permet de mesurer l'erreur lorsque le seuil est fixé de façon optimale. Par abus de notation, on notera ces métriques $minDCF$ et $minAPD$.

$$minDCF = \inf_{\sigma\in\mathbb{R}} C^\sigma_{avg} \quad minAPD = \inf_{\sigma\in\mathbb{R}} APD^\sigma$$

**Moyennes par langue cible**

Les métriques $EER$, $minDCF$ et $minAPD$, introduites pour un système de détection de plusieurs langues, mesurent la performance du système en éliminant une partie de l'erreur de calibration due au choix du seuil $\sigma$. Elles imposent cependant que ce seuil $\sigma$ soit le même pour toutes les langues cibles. Pour un système réel, pour lequel les scores ne sont pas parfaitement calibrés, le seuil optimal n'est pas le même pour toutes les langues cibles. On peut relâcher cette contrainte et calculer un taux d'égale erreur pour chaque langue cible, puis réaliser la moyenne des valeurs. Cette métrique est appelée la moyenne du taux d'égale erreur et est notée $EER_{avg}$ [Lozano-Diez *et al.*, 2018a].

Pour une langue cible $L_\mathcal{T}$ et pour l'ensemble des langues non cibles associé $\mathcal{L}_\mathcal{N}(L_\mathcal{T})$, nous pouvons définir des taux de faux rejet et de fausse acceptation.

$$P_{FR}(L_\mathcal{T}) = \frac{FN(L_\mathcal{T})}{VP(L_\mathcal{T}) + FN(L_\mathcal{T})}$$

$$P_{FA}(L_\mathcal{T}) = \frac{1}{|\mathcal{L}_\mathcal{N}(L_\mathcal{T})|} \sum_{L_\mathcal{N}\in\mathcal{L}_\mathcal{N}(L_\mathcal{T})} \frac{FP(L_\mathcal{T},L_\mathcal{N})}{VN(L_\mathcal{T},L_\mathcal{N}) + FP(L_\mathcal{T},L_\mathcal{N})}$$

Ces valeurs dépendent du seuil $\sigma$ de détection de la langue $L_\mathcal{T}$. On les notera $P^\sigma_{FR}(L_\mathcal{T})$ et $P^\sigma_{FA}(L_\mathcal{T})$.

Le taux d'égale erreur pour la langue $L_{\mathcal{T}}$ correspond au point de fonctionnement pour lequel ces deux taux d'erreur sont égaux.

$$EER(L_{\mathcal{T}}) \underset{P^{\sigma}_{FR}(L_{\mathcal{T}})=P^{\sigma}_{FA}(L_{\mathcal{T}})}{=} P^{\sigma}_{FR}(L_{\mathcal{T}})$$

La métrique $EER_{avg}$ est la moyenne arithmétique des taux d'égale erreur pour l'ensemble des langues cibles.

$$EER_{avg} = \frac{1}{|\mathcal{L}_{\mathcal{T}}|} \sum_{L_{\mathcal{T}} \in \mathcal{L}_{\mathcal{T}}} EER(L_{\mathcal{T}})$$

### 2.2.5 Point de vue sur les métriques

Les métriques d'évaluation ont deux rôles fondamentaux : permettre une comparaison objective des systèmes et rendre compte des caractéristiques de ceux-ci.

Afin de positionner nos systèmes par rapport à la littérature, il est indispensable d'utiliser les ensembles d'évaluation standards de la communauté ainsi que les métriques associées. En fonction des ensembles d'évaluation, celles-ci sont $C_{avg}$ ou $APD$ pour des points de fonctionnement particuliers. Si la recherche de la meilleure performance sur un corpus d'évaluation donné ne doit pas être le seul but de la recherche en apprentissage automatique, l'obtention de performances comparables ou supérieures à la littérature est une garantie de la bonne implémentation des systèmes qui conforte les résultats obtenus.

D'autre part, dans le but de mieux cerner certaines caractéristiques des systèmes, il peut être souhaitable d'utiliser d'autres métriques. Nous ne nous sommes pas penchés longuement sur l'étape de calibration des scores. Nous utilisons donc souvent des métriques rendant compte de la capacité de discrimination des scores, en faisant abstraction d'une partie de l'erreur de calibration : $EER$, $EER_{avg}$, $minDCF$, $minAPD$.

## 2.3 Positionnement par rapport à la littérature

La tâche de reconnaissance de la langue est étudiée depuis les années 1970 [House et Neuburg, 1977]. Sans prétendre rendre compte de façon exhaustive des recherches menées depuis lors, nous donnons quelques repères historiques dans le but de motiver les choix de systèmes utilisés aujourd'hui. L'évolution des systèmes de reconnaissance de la langue suit de près la progression des systèmes de reconnaissance du locuteur au cours des dernières décennies [Snyder, 2020, Cai *et al.*, 2018c].

### 2.3.1 Références

Haizhou Li a publié une revue de littérature sur la reconnaissance de la langue, avant l'établissement de la domination des réseaux de neurones dans les systèmes [Li *et al.*, 2013]. Notre travail se place dans la continuité de plusieurs thèses récentes. Celle de Niko Brümmer [Brümmer, 2010] définit le cadre d'évaluation des systèmes et la notion de calibration. Celle de Florian Verdet [Verdet, 2011] étudie l'analyse factorielle comme méthode

de réduction de la variabilité pour les systèmes de reconnaissance de la langue. La thèse de Grégory Gelly [Gelly, 2017] se concentre sur les réseaux de neurones récurrents. Alicia Lozano-Diez [Lozano-Diez, 2018] a étudié les différentes modalités d'emploi des réseaux de neurones profonds dans les systèmes de reconnaissance de la langue : identification de la langue de bout en bout, utilisation de *bottleneck features* et extraction d'*embeddings* de reconnaissance de la langue. Enfin, les systèmes dits *x-vectors* qui constituent l'état de l'art dans les années 2018 à 2021 ont été introduits au cours de la thèse de David Snyder [Snyder, 2020].

### 2.3.2 Repères historiques

La tâche de reconnaissance de la langue a historiquement été abordée avec deux types d'approches : phonotactiques et acoustiques. Les systèmes à l'état de l'art en 2021 obéissent à des approches acoustiques.

Les approches phonotactiques s'appuient sur une modélisation de l'enchaînement des sons dans une langue. Ces approches reviennent à reconnaître la suite des sons prononcés à partir du signal audio et à attribuer une vraisemblance à cette séquence pour chacune des langues visées. Une telle approche est appelée *Phone Recognition Followed by Language Modeling* [Glembek *et al.*, 2008]. Lorsque plusieurs systèmes de reconnaissance de phones (phones de différentes langues) sont utilisés, on parle de *Parallel Phone Recognition Followed by Language Modeling* [Zissman, 1996]. La vraisemblance de chaque langue est souvent calculée avec un modèle *n-gram* caractérisant l'enchaînement des phones. L'entraînement du système de reconnaissance de phones nécessite des ressources linguistiques : un alphabet phonétique, éventuellement un dictionnaire de prononciation et des corpus d'enregistrements transcrits pour la langue visée.

À l'inverse, les approches acoustiques visent à prédire la langue directement à partir des caractéristiques physiques du signal audio. Elles se basent souvent sur une représentation spectrale du signal. Les premières approches fonctionnaient à l'échelle d'une trame. Les scores par trame étaient ensuite moyennés afin de produire un score pour l'ensemble du segment. Les *Gaussian Mixture Models* [Zissman, 1996, Singer *et al.*, 2003] (*GMM*, mélanges de modèles Gaussiens) ont longtemps été les modèles dominants pour cette tâche.

Une rupture importante a été l'établissement du concept d'*embedding* : un unique vecteur caractérisant l'ensemble du segment et contenant l'information utile à la tâche. Ce vecteur est ensuite traité par un module appelé classifieur final (*back-end*) qui doit produire les scores de reconnaissance de la langue.

Le premier type d'*embedding* utilisé a été le supervecteur (*supervector*) des moyennes d'un modèle *GMM* [Campbell *et al.*, 2006]. Un modèle universel de représentation des trames acoustiques est appris et appelé *Univeral Background Model* (*UBM*). Les moyennes des composantes de l'*UBM* sont ensuite adaptées pour la distribution des trames du segment de test avec la méthode du *maximum a posteriori* (*MAP*) et concaténées pour former le supervecteur. Cette représentation a une dimension très importante.

Des méthodes de réduction de la dimension ont donc été introduites : l'analyse factorielle [Verdet, 2011] et le *nuisance attribute projection* [Campbell *et al.*, 2008]. Ces

méthodes visent à sélectionner un espace de représentation des supervecteurs qui élimine les sources de variabilité non utiles à la reconnaissance de la langue (parfois appelées facteurs de nuisance) comme la session ou le canal de transmission. La méthode appelée *joint factor analysis* [Kenny, 2005] consiste en l'estimation à partir du supervecteur de deux variables latentes représentant les informations de session et de langue. La méthode de la *total variability* [Dehak *et al.*, 2011] a permis de sélectionner un espace contenant l'ensemble de la variabilité portée par le supervecteur, information de langue comprise. L'application de cette méthode a conduit au développement du modèle d'*embedding* appelé *i-vector* [Dehak *et al.*, 2011, Martinez *et al.*, 2011] qui a dominé les systèmes de reconnaissance de la langue pendant la première moitié des années 2010 [Sturim *et al.*, 2016]. Les systèmes utilisés aujourd'hui conservent la structure des systèmes *i-vectors* avec trois modules principaux : extraction de *features* à l'échelle d'une trame, extraction d'un *embedding* pour l'ensemble du segment et classification finale.

La rupture suivante pour ces systèmes est l'apport des réseaux de neurones profonds. Une première direction de recherche a consisté à remplacer certains modules du système *i-vectors* par des réseaux de neurones. Utilisés afin d'extraire des *bottleneck features* [Matějka *et al.*, 2014, Jiang *et al.*, 2014], représentations de l'information phonétique contenue dans une trame, les réseaux de neurones profonds ont permis d'établir un pont entre les approches phonotactiques et acoustiques.

Un réseau de neurones peut également réaliser directement la tâche de reconnaissance de la langue. Certains travaux réalisent cette classification à l'échelle de la trame, puis agrègent les scores pour produire une prédiction pour l'ensemble du segment [Gonzalez-Dominguez *et al.*, 2015]. D'autres systèmes proposent de les utiliser directement pour réaliser la tâche de reconnaissance de la langue pour l'ensemble du segment. Le réseau doit alors être doté d'un mécanisme permettant de prendre en compte des segments de durée variable. Un tel modèle peut produire directement des prédictions. On parle alors de reconnaissance de la langue *de bout en bout* (*end-to-end*) [Lozano-Diez *et al.*, 2015, Cai *et al.*, 2018b]. Un *embedding* peut également être extrait d'une couche cachée du réseau, pour représenter le segment [Lozano-Diez *et al.*, 2018a]. On parle de *x-vectors* [Snyder *et al.*, 2018]. Les *x-vectors* ont remplacé les *i-vectors* comme système à l'état de l'art, à partir de la campagne *NIST LRE 2017* [Sadjadi *et al.*, 2018a].

## 2.4 Description des systèmes à l'état de l'art

Nous nous concentrons ici sur les systèmes basés sur un réseau de neurones d'identification de la langue. Ils constituent depuis 2017 le type de système dominant pour la tâche de reconnaissance de la langue et c'est sur eux qu'a porté notre travail.

### 2.4.1 Architecture générale

L'architecture d'un système de reconnaissance de la langue est représentée en figure 2.1. Un tel système est constitué de trois blocs principaux. Un premier bloc est chargé d'extraire des *features* à l'échelle d'une trame. Un second bloc, constitué d'un réseau

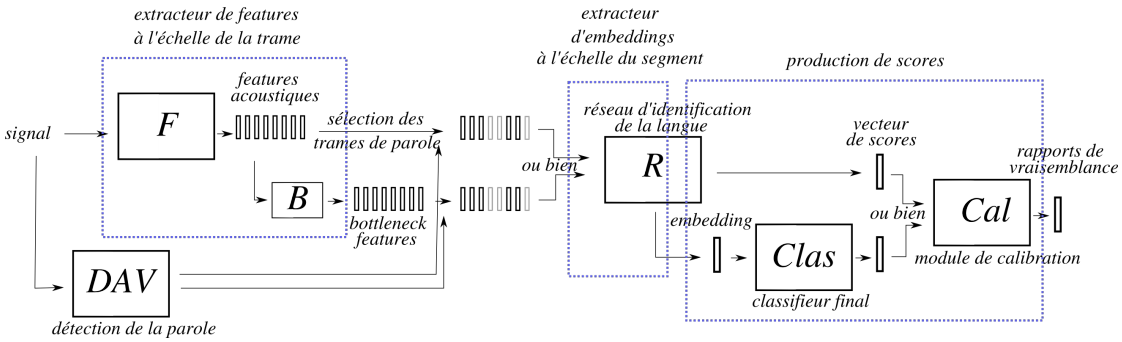



FIGURE 2.1 – Architecture d'un système de reconnaissance de la langue, constitué de plusieurs modules : l'extracteur de *features* acoustiques $F$, l'extracteur de *bottleneck features* $B$, le détecteur d'activité vocale $DAV$, le réseau d'identification de la langue $R$, le classifieur final $Clas$ et le module de calibration $Cal$.

de neurones, extrait un *embedding* caractérisant l'ensemble du signal d'entrée. Enfin un classifieur produit un score pour chaque hypothèse à partir de cet *embedding*. Notons que cette architecture est identique aux systèmes *i-vectors*, la seule différence étant que l'extracteur d'*embeddings* est alors un modèle *i-vectors* et non un réseau de neurones et qu'il ne peut être employé pour produire directement des scores.

Nous explicitons dans un premier temps le rôle de chacun des modules et leur articulation au sein d'un système. Ensuite, nous passons en revue les principaux choix d'architecture existant pour chacun des modules.

### Rôle des modules et notations

Soit $\mathcal{X}$ l'ensemble des signaux traités par le système $s$. Un enregistrement audio $x$ est un signal monodimensionnel digital. Il est échantillonné dans le temps selon une fréquence d'échantillonnage $\mathcal{F}_S$ exprimée en Hertz (Hz). Pour la tâche de reconnaissance de la langue, on considérera souvent des fréquences d'échantillonnage de 8000 Hz (téléphone fixe) ou 16000 Hz. Un signal $x$ de durée $T$ exprimée en secondes (s) est donc constitué de $T_S = T \times \mathcal{F}_S$ échantillons. Nous nous plaçons dans le cadre d'évaluation défini dans la partie précédente. Le système $s$ est un extracteur de scores chargé de produire un rapport de vraisemblances pour chacune des hypothèses de l'ensemble $\mathcal{Y}$.

Les systèmes de reconnaissance de la langue font souvent l'hypothèse que le signal d'entrée contient uniquement de la parole. Un premier module est donc le module de détection d'activité vocale $DAV$. Ce module est chargé de produire une segmentation du signal d'entrée $x$ correspondant aux moments où le signal contient de la parole. Dans la suite des traitements, seuls les segments contenant de la parole seront utilisés pour réaliser la prédiction.

L'extracteur de *features* acoustiques est noté $F$. Les *features* acoustiques font intervenir la notion de trame. Une trame est une portion de signal. $F$ produit un vecteur de dimension notée $d_F$ chargé de représenter cette portion de signal. La représentation par

trames est caractérisée par deux paramètres : la durée d'une trame $\Delta_F$ et l'espacement entre deux trames successives $\delta_F$. En reconnaissance de la langue, les durées typiquement utilisées sont de 20 à 30 millisecondes pour la durée $\Delta_F$ et un espacement $\delta_F$ de 10 millisecondes. Cela signifie qu'il y a un recouvrement entre deux trames successives. La séquence de *features* acoustiques produite a une longueur $T_F$ proportionnelle à la durée $T$ du signal d'entrée. La fréquence $\mathcal{F}_F$ des *features* acoustiques correspond à l'inverse de la durée séparant deux trames. Elle vaut en général 100 Hertz.

L'extracteur de *bottleneck features* $B$ a vocation à raffiner l'information contenue dans les vecteurs acoustiques afin de les rendre plus adaptés à la tâche de reconnaissance de la langue. À partir de la séquence de *features* acoustiques d'entrée, il produit une séquence de *bottleneck features* de dimension $d_B$ et avec une longueur $T_B$ correspondant à une fréquence $\mathcal{F}_B$, éventuellement inférieure à $\mathcal{F}_F$.

Le réseau de neurones d'identification de la langue prend en entrée une séquence de vecteurs caractérisant le signal et est chargé de produire une représentation de taille fixe du signal, indépendamment de la durée du signal d'entrée. Quand cette représentation est directement un vecteur de scores pour les différentes hypothèses de l'ensemble $\mathcal{Y}$, en général des probabilités *a posteriori*, ce module sera noté $R$. Si ce réseau est utilisé pour extraire un vecteur de dimension $d_e$, représentant le signal, nous le noterons $R_e$. Un tel vecteur est appelé *embedding* (plongement).

Lorsqu'un *embedding* $e$ est extrait du réseau $R_e$, un module appelé classifieur final et noté $Clas$ est responsable de produire des scores pour chaque hypothèse à partir de cet *embedding*.

Enfin, il est courant d'appliquer une transformation sur les scores afin de leur donner une interprétation probabiliste. Cette opération est réalisée par un module de calibration $Cal$. Elle vise à produire des rapports de vraisemblance pour chaque hypothèse de $\mathcal{Y}$.

#### Enchaînement des modules

Les différents modules sont appliqués sur le signal dans l'ordre dans lequel nous les avons énoncés : extraction de *features* acoustiques, de *bottleneck features*, extraction d'un *embedding*, production de scores et calibration. Notons que certains modules sont optionnels. Les *features* acoustiques peuvent être directement fournis au réseau d'identification de la langue, sans passer par des *bottleneck features*. De même, le réseau d'identification de la langue peut être directement utilisé pour produire des scores, sans classifieur final. Il est également possible de s'abstenir de module de détection d'activité vocale, en le remplaçant par une couche d'attention chargée d'éliminer certaines trames.

Une subtilité doit être mentionnée concernant la segmentation produite par le détecteur d'activité vocale. Il est utilisé pour sélectionner une sous-séquence de la séquence des *features* avant de la fournir au réseau d'identification de la langue. En règle générale, cette sélection n'est pas appliquée avant le calcul des *features* ou *bottleneck features* car ceux-ci utilisent un contexte autour de chaque trame, même s'il contient uniquement du bruit ou du silence et pas de parole.

Notons également qu'il est envisageable d'utiliser plusieurs versions de chacun des modules. Ainsi plusieurs séquences de *features* acoustiques peuvent être extraites et conca-

ténées [Li *et al.*, 2020c, Li *et al.*, 2020b] , les *bottleneck features* peuvent également être concaténés aux *features* acoustiques. Plusieurs *embeddings* peuvent être extraits à partir d'un même signal et concaténés [Li *et al.*, 2020b]. On peut également fusionner plusieurs sous-systèmes parallèles, qui extraient chacun un vecteur de scores à partir du signal d'entrée. Le module de calibration est en général responsable de réaliser la fusion de ces vecteurs de scores pour produire un rapport de vraisemblance pour chaque hypothèse.

Enfin remarquons que les modules peuvent être intégrés. Comme nous l'avons mentionné, dans de nombreux cas le réseau de neurones central n'est pas utilisé pour extraire un *embedding* mais directement pour produire des scores. L'extracteur d'*embeddings* et le classifieur final sont alors intégrés et on parle de système *end-to-end* (de bout en bout) [Trong *et al.*, 2016, Monteiro *et al.*, 2019, Wan *et al.*, 2019]. Certains travaux soulignent de meilleures performances obtenues avec un réseau *end-to-end* qu'en utilisant un classifieur final [Lopez *et al.*, 2018]. D'autres observent le contraire et mettent en avant le fait qu'un classifieur final présente plus de flexibilité : il peut être entraîné sur un autre ensemble de langues que celles ayant servi à l'apprentissage de l'extracteur d'*embeddings* [Snyder *et al.*, 2018]. En reconnaissance du locuteur, il a été proposé d'intégrer également le module de calibration dans le réseau de neurones central [Garcia-Romero *et al.*, 2020]. Dans le chapitre 6 de cette thèse, nous nous intéressons à l'intégration de l'extracteur de *bottleneck features* dans le réseau de neurones central. Enfin, certains systèmes de traitement de la parole proposent également d'intégrer le module d'extraction de *features* acoustiques dans le réseau de neurones central [Ravanelli et Bengio, 2018, Zeghidour *et al.*, 2020], même si ce n'est pas la norme en reconnaissance de la langue.

### 2.4.2 *Features* acoustiques

Le premier module d'un système de reconnaissance de la langue est chargé de l'extraction de *features* acoustiques. Ces *features* constituent une représentation du signal supposée présenter de façon plus accessible l'information contenue dans le signal d'entrée. Chaque trame de durée $\delta_F$ est représentée par un vecteur de dimension $d_F$.

Les *features* acoustiques les plus classiquement utilisés sont basés sur un spectrogramme [Cai *et al.*, 2020b], c'est-à-dire une représentation temps-fréquence. Des transformations peuvent être appliquées comme la projection en échelle *Mel* [Stevens *et al.*, 1937], inspirée d'une analyse de l'échelle de perception de l'oreille humaine. On utilise souvent le logarithme des spectrogrammes *Mel* [McLaren *et al.*, 2018b]. La transformée en cosinus discrète appliquée aux bancs de filtre en échelle *Mel* permet de décorréler les différentes composantes des vecteurs, dans le but de produire des représentations plus compactes et adaptées à un système de classification. Ces représentations sont appelées *Mel Frequency Cepstral Coefficients* (*MFCC*) [Davis et Mermelstein, 1980]. D'autres transformations du spectrogramme sont parfois employées, par exemple les *PLP* (*perceptual linear prediction*)*features* [Gelly et Gauvain, 2017, BenZeghiba *et al.*, 2012].

Enfin, les *features* peuvent être présentés aux modules suivants en prenant en compte leur variation temporelle. Par exemple, pour les *MFCC*, on ajoute souvent leur variation temporelle aux premier ($\Delta$) et second ordres ($\Delta\Delta$). Les dérivées premières et secondes

sont calculées en utilisant les trames voisines et simplement concaténées au vecteur de *features*. Lorsque les différences sont calculées en laissant un intervalle entre les trames, on parle de *Shifted Delta Coefficients* (*SDC*) [Torres-Carrasquillo *et al.*, 2002].

Une direction récente de recherche vise à apprendre les filtres utilisés pour extraire des *features* acoustiques, en les considérant comme des couches de réseaux de neurones [Ravanelli et Bengio, 2018, Zeghidour *et al.*, 2020].

### 2.4.3 Bottleneck features

Les *features* acoustiques peuvent être fournis directement au réseau d'identification de la langue. Cependant ces *features* peuvent également être raffinés afin de produire une représentation plus adaptée à la tâche de reconnaissance de la langue, ce qui améliore nettement les performances [Snyder *et al.*, 2018]. Un réseau de neurones est utilisé pour sélectionner l'information pertinente grâce à une tâche auxilaire [Ling *et al.*, 2020, Pascual *et al.*, 2019]. La méthode ayant donné les meilleurs résultats est appelée *bottleneck features* [Fér *et al.*, 2017] et est fondée sur une tâche de reconnaissance de la parole.

La reconnaissance de la parole repose en partie sur une information phonétique que nous savons être cruciale pour la tâche de reconnaissance de la langue [Li *et al.*, 2013]. Les systèmes de reconnaissance de la langue phonotactiques convertissent les signaux en séries de phones propres à chaque langue et calculent la vraisemblance de chaque série de phones selon un modèle de langage. À l'inverse, les méthodes acoustiques, fondées sur des représentations spectrales, n'utilisent pas d'information phonétique. Les *bottleneck features* incarnent un intermédiaire entre ces deux types d'approches. Ils permettent d'utiliser un modèle de reconnaissance de la parole afin de produire des représentations plus adaptées au sein d'un modèle fondé sur une approche strictement acoustique.

Les *bottleneck features* sont des représentations du signal à l'échelle d'une trame. Ils correspondent à l'activation d'une couche intermédiaire d'un réseau de neurones entraîné à prédire une étiquette de reconnaissance de la parole pour une trame d'entrée. En pratique la trame d'entrée est présentée au réseau avec un contexte de quelques trames. Les étiquettes de reconnaissance de la parole utilisées sont des monophones ou des triphones. Elles nécessitent un alignement forcé entre les trames et les étiquettes, et donc l'utilisation d'un modèle de reconnaissance de la parole. Différentes architectures ont été proposées pour réaliser cette tâche : des *Time-Delay Neural Networks* [Peng *et al.*, 2019], des réseaux récurrents [Tian *et al.*, 2016, Mateju *et al.*, 2018] et des perceptrons multi-couches [Silnova *et al.*, 2018]. Ils comportent une couche de dimension faible (par exemple 64 ou 80) dite *bottleneck* à partir de laquelle les *bottleneck features* sont extraits.

Notons que le processus de raffinement des représentations peut être réalisé plusieurs fois, *en cascade*. On peut par exemple entraîner un premier réseau avec un contexte assez faible, s'en servir pour extraire des *bottleneck features*, puis entraîner un second réseau pour la même tâche en prenant en entrée ces *bottleneck features* avec un contexte plus large. De nouveaux *bottleneck features* seront extraits de ce second réseau. On parle alors de *stacked bottleneck features* [Plchot *et al.*, 2018]. Remarquons enfin que des *bottleneck features* de meilleure qualité pour la tâche de reconnaissance de la langue peuvent être obtenus en utilisant plusieurs langues pour la tâche de reconnaissance de phones [Fér

*et al.*, 2017].

L'entraînement des réseaux d'extraction de *bottleneck features* repose sur un alignement trame à trame avec les étiquettes de phone. Cela est généralement réalisé par un alignement forcé à l'aide d'un modèle de reconnaissance de la parole par exemple basé sur une chaîne de Markov cachée (*Hidden Markov Model - HMM*) [Fér *et al.*, 2017]. Mais des modèles de reconnaissance de la parole avec un réseau de neurones de bout-en-bout ont été proposés récemment [Gulati *et al.*, 2020, Amodei *et al.*, 2016]. Certains travaux utilisent directement ces réseaux de bout en bout pour extraire des *bottleneck features* sans recourir à la phase d'alignement forcé [Ling *et al.*, 2020]. C'est aussi l'approche que nous avons adoptée pour le *challenge OLR 2020*, et nous la présentons dans le chapitre 6 de ce document.

### 2.4.4 Réseau de neurones d'identification de la langue

Depuis l'évaluation *NIST LRE 2017*, la tâche de reconnaissance de la langue est dominée par des systèmes dont l'élément central est un réseau de neurones entraîné à identifier la langue. Il prend en entrée une séquence de *features* (acoustiques ou *bottleneck features*) et est utilisé soit pour produire directement un score pour chaque hypothèse, soit pour produire un *embedding* supposé contenir l'information utile à la tâche de reconnaissance de la langue pour l'ensemble du segment. Pour faire un panorama de l'ensemble des modèles proposés, distinguons les architectures et les méthodes d'entraînement.

#### Architectures

Une contrainte majeure s'applique à l'architecture : elle doit produire une représentation de taille fixe (*embedding* ou vecteur de scores) à partir d'une séquence de *features* de durée arbitraire. Deux approches existent : des architectures prenant en entrée uniquement des séquences d'une durée fixée *a priori*, ou bien des architectures prenant en compte une durée variable.

**Architectures prenant en compte une durée fixe** À partir du moment où la durée de la séquence d'entrée est fixée, la tâche peut être réalisée par n'importe quelle architecture de réseau de neurones sous la condition de respecter les dimensions des vecteurs d'entrée et de sortie. Par exemple un réseau de neurones convolutionnel a été proposé pour des segments de trois secondes [Lozano-Diez *et al.*, 2015]. Un tel modèle ne peut être employé dans un cadre normal d'évaluation où les durées des segments varient. Dans [Lozano-Diez *et al.*, 2015], ce problème est résolu en réalisant du *padding* pour les segments plus courts.

**Architectures prenant en compte une durée variable** Un tel système prenant en compte une durée variable doit être capable d'agréger l'information selon l'axe temporel afin de produire une représentation de taille fixe. Il peut être décomposé en trois étapes : traitements conservant la dimension temporelle, agrégation temporelle et traitement global de l'ensemble du segment.

Remarquons qu'il est également possible de réaliser une prédiction de reconnaissance de la langue par trame, puis d'agréger les vecteurs de scores. C'est le cas par exemple dans [Zazo *et al.*, 2016b] avec un *LSTM* qui produit un vecteur de scores par trame. Le type d'architecture que nous présentons est plus général : les représentations par trame produites par le réseau ne sont pas nécessairement des scores de reconnaissance de la langue.

**Traitements conservant la dimension temporelle** Quatre types de couches conservent la dimension temporelle : couches récurrentes, convolutions monodimensionnelles, convolutions bidimensionnelles et auto-attention.

Les couches récurrentes [Gelly et Gauvain, 2017, Geng *et al.*, 2016] produisent une séquence en traitant récursivement la séquence d'entrée dans l'ordre temporel. Afin de prendre en compte un contexte des deux côtés de chaque trame, il est courant d'utiliser un réseau récurrent bidirectionnel [Gelly *et al.*, 2016, Lozano-Diez *et al.*, 2018b] (dans le sens du temps et dans le sens inverse) et de concaténer les représentations produites par les deux dimensions. Différentes paramétrisations des couches récurrentes ont été proposées : les *LSTM* [Zazo *et al.*, 2016a, Tang *et al.*, 2017b, Gelly et Gauvain, 2017, Zazo *et al.*, 2016b, Wan *et al.*, 2019] et les *GRU* [Trong *et al.*, 2016, Padi *et al.*, 2019b] par exemple.

Les couches convolutionnelles à une dimension consistent à appliquer un même filtre sur les trames d'entrée assorties de leur contexte et à le faire glisser selon l'axe temporel. Les convolutions à une dimension suivies de *pooling* ont la propriété de produire des représentations invariantes à une translation selon l'axe temporel, ce qui est souhaitable pour la reconnaissance de la langue. Parfois une dilatation est appliquée afin de prendre en compte un contexte plus large sans utiliser plus de paramètres. Cela consiste à ignorer des trames en appliquant par exemple un noyau de convolution de taille 3 aux trames $\{t-2,t,t+2\}$ au lieu des trames $\{t-1,t,t+1\}$. Des convolutions à une dimension avec dilatation sont parfois appelées *Time Delay Neural Network* (*TDNN*) [Snyder *et al.*, 2018]. Des raffinements des couches *TDNN* ont été proposés pour la reconnaissance du locuteur : *Factorized TDNN* et *Extended TDNN* [Villalba *et al.*, 2020], *ECAPA TDNN* [Desplanques *et al.*, 2020].

Les couches convolutionnelles à deux dimensions appliquent un filtre sur les entrées selon les dimensions temporelles et fréquentielles. Une telle approche est inspirée par des succès en traitement de l'image [Krizhevsky *et al.*, 2017]. Elle peut être directement appliquée à la reconnaissance de la langue en considérant les spectrogrammes comme des images [Sarthak et Mittal, 2019]. Ainsi les convolutions à deux dimensions sont parfaitement adaptées à des spectrogrammes structurés selon un axe fréquentiel [Cai *et al.*, 2018c], selon lequel on peut espérer détecter des motifs invariants par translation. En revanche l'invariance par translation selon l'axe des *features* n'est pas attendue pour des *MFCC*, *SDC-MFCC* [Lozano-Diez *et al.*, 2018a] ou *bottleneck features* pour lesquels les différentes composantes sont supposées décorrélées.

Enfin les couches d'auto-attention permettent de préserver une représentation locale tout en modélisant des dépendances sur l'ensemble du segment [Vaswani *et al.*, 2017]. Ces

couches sont souvent appelées *transformer*. Elles ont été employées en reconnaissance du locuteur [Shi *et al.*, 2020].

Les couches peuvent être enchaînées, parfois même en mêlant des couches de natures différentes, par exemple les architectures *CRNN* (*Convolutional Recurrent Neural Network*), *CNN-LSTM-TDNN* [Miao *et al.*, 2019], *CNN* et *GRU* [Trong *et al.*, 2016], *CNN-BLSTM* [Cai *et al.*, 2019]. Les couches sont séparées par des non-linéarités et éventuellement par des couches de normalisation utiles à la stabilité de l'entraînement. Au lieu d'être simplement enchaînées, les couches peuvent être utilisées de façon plus structurées. Par exemple les connections résiduelles consistent en l'ajout à la sortie d'une couche des activations d'une couche précédente dans le but de faciliter la retropropagation du gradient [Monteiro *et al.*, 2019, Villalba *et al.*, 2020].

**Agrégation temporelle** La première partie du réseau de neurones produit des représentations portées par un axe temporel, éventuellement de dimension variable correspondant à la durée. Le module d'agrégation selon l'axe temporel a pour rôle de produire une représentation de dimension fixe, représentant l'ensemble du segment et dépouillée de la dimension temporelle. Deux types de méthodes ont été proposés : un réseau récurrent et l'accumulation statistique.

Par définition, une couche récurrente traite récursivement l'ensemble de la séquence d'entrée (éventuellement selon les deux directions). Elle permet donc d'extraire une représentation de taille fixe, en prenant la sortie de la couche après avoir traité l'ensemble de la séquence [Heigold *et al.*, 2016].

Une autre manière naturelle de rendre compte d'une séquence de vecteurs à travers une représentation de taille fixe est l'accumulation de statistiques sur la séquence. Cette idée a été réalisée en calculant les premiers moments de la distribution de la séquence, composante par composante. Ainsi ont été proposées une simple moyenne, appelée *temporal average pooling* [Cai *et al.*, 2018c], et la concaténation de la moyenne et de l'écart-type, appelée *statistical pooling* (*pooling* statistique) [Snyder *et al.*, 2018, Novotný *et al.*, 2018]. Un raffinement de cette idée consiste à accumuler des statistiques sur les distances à une famille de vecteurs appelée dictionnaire et apprise au cours de l'entraînement. Il s'agit de la couche appelée *Learnable Dictionary Encoding* (*LDE*) [Cai *et al.*, 2018b]. En reconnaissance du locuteur, a été étudié l'ajout de moments d'ordre supérieur de la distribution de la séquence : coefficient d'asymétrie et kurtosis [Rouvier *et al.*, 2021a].

Quelle que soit la méthode d'agrégation, elle peut être combinée avec un mécanisme d'attention [Monteiro *et al.*, 2019]. Celui-ci permet d'attribuer un poids à chaque trame dans l'agrégation en fonction de son intérêt pour la tâche utilisée pour l'entraînement du réseau. Un mécanisme d'attention peut également être appliqué sur la dimension fréquentielle [Miao *et al.*, 2019].

Des comparaisons de la performance de plusieurs couches d'agrégation statistique ont été effectuées dans [Cai *et al.*, 2018c] en reconnaissance de la langue et dans [Villalba *et al.*, 2020, Wang *et al.*, 2021b, Rouvier *et al.*, 2021a] en reconnaissance du locuteur. Elles mettent en avant l'importance de l'écart-type et l'intérêt de la couche *learnable dictionary encoding*.

**Traitement global** Après l'étape d'agrégation temporelle, le segment est représenté par un unique vecteur de dimension fixe. Les couches suivantes consistent en un traitement de ce vecteur dans le but de produire un *embedding* ou un vecteur de scores pour chaque langue, souvent les deux séquentiellement. Ce traitement est effectué par un perceptron multi-couches avec des couches linéaires entrecoupées de non-linéarités [Snyder *et al.*, 2018]. C'est souvent ce perceptron multi-couches qui est le siège des couches de *dropout* lorsqu'elles sont utilisées au cours de l'optimisation.

**Architecture de référence** Les paradigmes généraux de définition des architectures des réseaux d'identification de la langue ont été exposés. Les choix des hyperparamètres de ces architectures sont réalisés empiriquement en fonction des performances de reconnaissance de la langue obtenues sur des corpus de référence. En dépit de la diversité des architectures présentées, il existe un modèle de référence, employé par une large partie de la littérature [McLaren *et al.*, 2018a, Rouvier *et al.*, 2021b]. Il s'agit d'un *TDNN* avec une couche de *pooling* statistique [Snyder *et al.*, 2018].

### Méthode d'entraînement

L'algorithme incontournable pour l'entraînement d'un réseau de neurones est la descente de gradient stochastique [Bottou, 2012]. Les réseaux d'identification de la langue ne font pas exception.

Après une initialisation aléatoire des paramètres du réseau $\theta_0$, on échantillone séquentiellement des *minibatchs* $m_i$ et on minimise la fonction de coût $\mathcal{L}$ sur ces échantillons. La règle de mise à jour des paramètres $\theta$ lors de l'itération $i$ est la suivante :

$$\theta_{i+1} = \theta_i - \eta_i g_i$$

où $g_i = \nabla \mathcal{L}_\theta(m_i, \theta_i)$ est le gradient de la fonction de coût par rapport aux paramètres du réseau, estimé sur le *minibatch* $m_i$. $\eta_i$ est le pas de la descente de gradient. L'algorithme standard utilise un pas fixe ($10^{-2}$ ou $10^{-3}$ par exemple). Des raffinements de cet algorithme existent, par exemple en rajoutant un moment, une pénalisation de la norme des poids $\theta$, ou en estimant la courbure de la fonction de coût comme dans la méthode *Adam* [Kingma et Ba, 2017].

Les *minibatchs* sont échantillonnés sur des durées courtes, typiquement trois secondes [Lozano-Diez *et al.*, 2015]. Il est courant de raffiner ensuite le modèle obtenu par un premier entraînement sur des durées courtes avec des durées plus longues [Garcia-Romero *et al.*, 2020].

**Augmentation de données** Des augmentations de données peuvent être appliquées [Lozano-Diez *et al.*, 2018b, Novotný *et al.*, 2018, Richardson *et al.*, 2018, Mccree *et al.*, 2018]. Celles-ci consistent en des transformations des signaux dans le but d'augmenter la diversité des données d'entraînement et donc la généralisation du modèle. Des travaux montrent qu'un réseau de neurones d'identification de la langue est particulièrement sensible aux augmentations de données, en comparaison avec des *i-vectors* ou des

classifieurs finaux [Snyder *et al.*, 2018]. De nombreuses augmentations de données ont été proposées, généralement dans le but d'imiter des distorsions attendues du signal de parole. Parmi les méthodes d'augmentation de données largement utilisées, on compte l'ajout de bruits additifs, de *babble noise* (mélange de voix de plusieurs autres locuteurs) ou de musique, la réverbération afin de simuler une pièce à partir d'une base de données de réponses impulsionnelles, la variation de la vitesse, l'application de *codecs* [McLaren *et al.*, 2018b]. Une autre méthode d'augmentation consiste à masquer des parties du spectrogramme d'entrée selon des bandes temporelles ou fréquentielles. Elle est similaire à un *dropout* structuré sur la couche d'entrée et s'apparente donc à une méthode de régularisation. Il s'agit de *specAugment* [Park *et al.*, 2019].

**Fonction de coût** La fonction de coût est le critère minimisé par l'algorithme d'optimisation afin de sélectionner les paramètres du réseau. Si le réseau est utilisé directement pour produire des scores de reconnaissance de la langue, il semble naturel d'utiliser des métriques caractérisant la performance du système ($EER$, $C_{avg}$, *etc.*). Cela n'est pas réalisable en raison de l'algorithme d'optimisation utilisé (descente de gradient) qui impose que la fonction de coût soit différentiable. Il faut donc utiliser d'autres fonctions de coût qui peuvent être réparties en deux catégories selon leur rôle. Les fonctions de coût de classification opèrent sur les scores et évaluent leur capacité à réaliser la tâche d'identification de la langue. Les fonctions de coût de *metric learning* opèrent sur un espace d'*embeddings*. Elles ont pour but d'imposer que les distances entre échantillons dans l'espace des *embeddings* correspondent à la répartition en différentes classes.

La fonction de coût par excellence de toutes les tâches de classification est l'entropie croisée [Ramos *et al.*, 2018], également appelée *softmax loss*. Des raffinements de l'entropie croisée ont été introduits dans l'objectif d'imposer une meilleure séparation entre les classes : *additive margin softmax* ($AM$) [Li *et al.*, 2020c], *additive angular margin softmax* ($AAM$) [Villalba *et al.*, 2020]. La *tuplemax loss* [Wan *et al.*, 2019] est une généralisation de l'entropie croisée qui prend en compte toutes les paires de langues.

Les fonctions de coût de *metric learning* ont été largement utilisées pour la reconnaissance du locuteur [Chung *et al.*, 2020] : la *triplet loss* [Chung *et al.*, 2020, Zhang et Koishida, 2017, Zhang *et al.*, 2018, Bahmaninezhad *et al.*, 2021], *prototypical networks* [Chung *et al.*, 2020] et une fonction de coût dérivée du modèle *PLDA* [Snyder *et al.*, 2016]. En reconnaissance de la langue, la *triplet loss* a été utilisée pour entraîner un classifieur final [Mingote *et al.*, 2019] et la *cosine similarity* ou la proximité angulaire [Gelly et Gauvain, 2017] pour entraîner un extracteur d'*embeddings* basé sur un *LSTM*. Comme l'évaluation des fonctions de coût de *metric learning* repose sur l'échantillonnage de paires de segments, des stratégies peuvent être développées afin d'améliorer l'efficacité de l'apprentissage. Par exemple, on peut favoriser l'échantillonnage de segments sur lesquels le modèle est peu performant : on parle de *negative mining* [Mingote *et al.*, 2019].

Les fonctions de coût de *metric learning* sont parfois utilisées en combinaison avec l'entropie croisée, comme par exemple la *center loss* [Cai *et al.*, 2018c] et la *triplet loss* [Monteiro *et al.*, 2019]. Nous explorons cette approche dans le chapitre 4.

**Sélection de l'ensemble final des paramètres** Une fois la descente de gradient terminée, quels paramètres choisir ? Certains prennent directement le dernier modèle [Cai *et al.*, 2018c]. Pour éviter le sur-apprentissage, la majorité des travaux sélectionne le modèle associé à la meilleure performance sur l'ensemble de validation. Une méthode introduite pour augmenter la généralisation appelée *Stochastic Weight Averaging* [Izmailov *et al.*, 2018] a été appliquée à la reconnaissance de la langue au cours de cette thèse. Elle revient à faire la moyenne des paramètres correspondant à différentes étapes de l'apprentissage.

L'entraînement d'un réseau de neurones par descente de gradient est un compromis entre deux objectifs : la maximisation de la performance sur l'ensemble d'entraînement et la régularisation afin de favoriser la généralisation. De nombreux éléments concourent à la régularisation : valeurs du pas élevées dans la descente de gradient, augmentations de données importantes, segments courts, *weight decay*, *early stopping*, *dropout*, larges marges dans les fonctions de coût. Le choix d'une recette d'apprentissage est un savoir-faire empirique consistant à sélectionner un dosage performant des différentes méthodes de régularisation pour une configuration donnée.

### 2.4.5 Classifieur final

Lorsque le réseau d'identification de la langue est utilisé pour extraire un *embedding*, la production de scores de reconnaissance de la langue est assurée par un classifieur final. Les classifieurs finaux sont construits sur le modèles de ceux qui avaient été développés pour les systèmes *i-vectors*. On procède d'abord généralement à une normalisation des *embeddings* constituée des étapes suivantes [Bousquet et Rouvier, 2019] : centrage par soustraction de la moyenne de la distribution, projection sur un sous-espace de dimension plus faible par une analyse discriminante linéaire (*linear discriminant analysis* - *LDA*, en anglais), blanchiment, la matrice de covariance de la distribution est ramenée à l'identité par multiplication matricielle, normalisation de la norme $L_2$ [Garcia-Romero et Espy-Wilson, 2011].

L'*embedding* normalisé est ensuite fourni en entrée d'un classifieur *back-end* qui peut être un modèle génératif ou discriminant. Le classifieur Gaussien est un modèle génératif qui consiste à représenter chaque classe par une distribution Gaussienne. Les vecteurs de moyenne et matrices de covariance de chaque distribution sont estimés sur l'ensemble d'entraînement. On impose souvent des contraintes sur les matrices de covariance : matrices diagonales ou bien une unique matrice de covariance commune à toutes les classes. Dans ce dernier cas, on parle de classifieur Gaussien linéaire [Lozano-Diez *et al.*, 2018b,Plchot *et al.*, 2018]. Lors de l'inférence, un score pour chaque langue est estimé en prenant la vraisemblance de l'*embedding* pour la Gaussienne de chaque classe. Notons qu'un classifieur Gaussien peut également être entraîné de façon discriminante [Mccree *et al.*, 2018]. Un exemple de modèle discriminant est une machine à vecteurs supports (*Support Vector Machine*, *SVM*) [Padi *et al.*, 2018]. Elle consiste à apprendre des hyperplans séparant les différentes classes, éventuellement dans un espace de représentations défini par un espace de Hilbert à noyau reproduisant. Les scores pour chaque langue sont ensuite définis en

fonction de la distance de l'*embedding* aux différents hyperplans. Des réseaux de neurones ont également été utilisés [Plchot *et al.*, 2016, Padi *et al.*, 2018].

### 2.4.6 Calibration

Le système de reconnaissance de la langue doit produire un score pour chacune des langues traitées par le système. La détection sera effectuée par l'application d'un seuil sur ce score. Dans le but d'être capable d'utiliser le même système dans différents contextes applicatifs, il est souhaitable de donner une interprétation probabiliste aux scores. Cette opération est appelée la calibration [Brümmer et Van Leeuwen, 2006]. De cette façon, le seuil pourra être directement déduit des paramètres du point de fonctionnement ($P_{Target}$,$C_{FA}$,$C_{Miss}$), comme expliqué dans la section 2.2. Les méthodes de calibration se limitent souvent à l'application d'une transformation affine de chaque score, dont les paramètres sont appris par une régression logistique [Martinez *et al.*, 2011, Brümmer, 2007] ou en minimisant l'entropie croisée. Un sujet important est la prise en compte des langues non vues dans l'ensemble d'entraînement par le module de calibration [McLaren *et al.*, 2017].

### 2.4.7 Fusion de systèmes

Lors des campagnes d'évaluation, les meilleures performances sont obtenues par la fusion de plusieurs systèmes. La fusion peut s'opérer à différents niveaux : partage de *features*, concaténation de plusieurs *embeddings* ou combinaison des scores. La combinaison des scores est la plus aisée à mettre en œuvre car elle permet le développement indépendant des sous-systèmes fusionnés. Le score final est une combinaison des scores des sous-systèmes. On se limite en général à une moyenne pondérée dont les paramètres sont appris au cours de l'étape de calibration [Jančík *et al.*, 2010].

### 2.4.8 Détection d'activité vocale

Les réseaux de neurones d'identification de la langue sont supposés prendre en entrée un signal de parole. Il convient donc d'appliquer un pré-traitement aux signaux d'entrée afin de ne fournir au système que les segments correspondant à de la parole. Cette étape est appelée *voice activity detection* (*VAD*) ou *speech activity detection* (*SAD*). Une segmentation imprécise de la parole impacte les performances, comme mesuré pour un système de reconnaissance du locuteur [McLaren *et al.*, 2018a] et un système de reconnaissance de la langue [Silnova *et al.*, 2018].

Le module de détection de la parole est en général découplé des autres modules du système. Il attribue chaque trame du signal à une des deux classes *parole* et *non-parole* et les trames de parole sont ensuite concaténées avant d'être présentées au réseau d'identification de la langue. Deux familles de modèles sont utilisées pour assigner une trame à une des classes : les modèles basés sur l'énergie et les réseaux de neurones.

Un modèle basé sur l'énergie calcule l'énergie du signal pour chaque trame et assigne la classe parole aux trames aux énergies les plus élevées [Mccree *et al.*, 2018]. C'est un modèle

assez simple qui fonctionne mieux dans les environnements peu bruités. Des perceptrons [Plchot *et al.*, 2018], réseaux de neurones récurrents [Richardson *et al.*, 2018, Mccree *et al.*, 2018] et convolutionnels [Thomas *et al.*, 2014] ont été entraînés pour la détection de la parole. L'entraînement d'un réseau de neurones nécessite un corpus annoté avec la segmentation, ce qui n'est pas toujours possible dans le cadre des campagnes d'évaluation qui imposent les ressources utilisées pour le développement du système. Pour certaines expériences, nous avons utilisé un réseau de neurones convolutionnel pour réaliser la détection d'activité vocale. Il est décrit en annexe C.

### 2.4.9 Point de vue sur les architectures de système

L'étude de la littérature sur les architectures des systèmes de reconnaissance de la langue force deux constats. Apparaît d'abord une diversité des choix possibles pour chaque module qui semble démultiplier le travail expérimental nécessaire pour comparer une nouvelle méthode aux approches existantes. D'autre part, des tendances de fond émergent : le remplacement progressif de chaque module par des réseaux de neurones profonds et l'intégration des différents modules du système dans un même modèle.

Nos travaux ne visent pas à l'amélioration intrinsèque de la performance des systèmes de reconnaissance de la langue mais à l'amélioration de leur robustesse au canal de transmission. Nous avons eu à cœur d'utiliser des architectures standards pour la plupart des modules et à proposer des méthodes d'augmentation de la robustesse qui ne seront pas rendues obsolètes dans un avenir proche par les deux dynamiques d'évolution des systèmes que nous avons identifiées. Nous avons donc uniquement travaillé sur des réseaux de neurones profonds : le réseau d'identification de la langue et l'extracteur de *bottleneck features* tout en étudiant leur possible intégration.

## 2.5 Directions actuelles de recherche

En 2021, la recherche en reconnaissance de la langue peut être répartie selon trois directions principales : l'amélioration des recettes d'apprentissage, le traitement de conditions difficiles et la segmentation.

### 2.5.1 Amélioration des recettes d'apprentissage

Une première lignée de travaux vise à simplifier la recette d'apprentissage. Pour les systèmes à l'état de l'art, elle est largement séquentielle : chaque module est entraîné après l'autre. Ceci pose la question de l'expertise nécessaire au développement d'un système et de l'optimalité globale de l'ensemble des modules. Certains travaux proposent le pré-entraînements de modules génériques : *multilingual bottleneck features* [Fér *et al.*, 2017] ou extracteur de *x-vectors* [Snyder *et al.*, 2018]. D'autres visent à proposer une approche d'apprentissage dite *de bout en bout* (*end-to-end*) dont tous les paramètres peuvent être appris conjointement [Lozano-Diez *et al.*, 2015].

Tous les modules sont entraînés par apprentissage supervisé [Ling *et al.*, 2020]. Ceci induit un coût d'annotation et limite les corpus d'apprentissage disponibles. De nom-

breux travaux étudient donc des méthodes d'apprentissage dites non supervisées ou semi-supervisées. Une approche prometteuse est le pré-entraînement de représentations non supervisées du signal [Baevski *et al.*, 2020].

### 2.5.2 Conditions difficiles

Au fur et à mesure que les performances dans les conditions étudiées s'améliorent, la communauté s'intéresse à des conditions d'évaluation plus difficiles. Ces conditions sont mises en lumière par les tâches proposées par les campagnes d'évaluation *NIST* et *OLR*. On évolue donc vers des segments courts (une seconde au lieu de trente [Shen *et al.*, 2018]). Des canaux de transmission nouveaux sont introduits : télévision, radio, VHF, au lieu de téléphonique. On s'intéresse également aux environnements bruités. De plus, les frontières entre les classes elles-mêmes sont affinées. Alors que de bonnes performances sont atteintes pour des langues très différentes, la reconnaissance de dialectes reste difficile. Certains travaux s'intéressent également à des locuteurs qui utilisent plusieurs langues [Titus *et al.*, 2020].

En plus de l'étude de conditions difficiles, la question de l'écart de certaines caractéristiques évoquées précédemment entre les corpus d'apprentissage et de test devient centrale. On parle de changement de domaine, notion qui est l'objet de cette thèse. L'état de l'art sur ce problème est exposé dans le chapitre 3.

### 2.5.3 Segmentation

Les performances des systèmes autorisent le déploiement dans des contextes multilingues [Punjabi *et al.*, 2021] et posent donc la question du changement de langue (*code switching*) et de la segmentation précise du signal (*language diarization*) par blocs de langue homogène [Lyu *et al.*, 2013].

### 2.5.4 Positionnement de nos travaux

Nos travaux s'inscrivent dans les deux premières directions de recherche. Nous visons des conditions difficiles : des canaux de transmission non vus ou peu vus dans l'ensemble d'apprentissage. Pour ce faire nous étudions des variations des méthodes d'entraînement des systèmes et notamment des méthodes semi-supervisées. D'autre part, nous montrons que l'entraînement du module d'extraction de *bottleneck features* peut être grandement simplifié, participant à l'amélioration des recettes d'apprentissage.

# 3

# Le canal de transmission : un domaine pour les systèmes de reconnaissance de la langue

Dans le chapitre 2, nous avons décrit les systèmes récents de reconnaissance de la langue. Les principaux modules de ces systèmes, extracteur de *bottleneck features*, extracteur d'*embeddings* et classifieur final, sont des modèles paramétriques, dont les paramètres sont sélectionnés par un algorithme appelé algorithme d'entraînement. Cet algorithme utilise un ensemble de données d'entraînement. Lors de l'utilisation du système les différents modules devront traiter d'autres données appelées données de test. L'art de la conception d'algorithmes d'entraînement est de permettre une bonne généralisation, c'est-à-dire une bonne performance sur les données de test alors même qu'elles sont plus ou moins différentes des données d'entraînement. Une telle généralisation est possible si le modèle utilise les données d'entraînement pour repérer certaines corrélations qui permettent de réaliser la tâche visée et que ces corrélations restent valides sur les données de test.

Les travaux en apprentissage automatique font souvent une hypothèse fondamentale : les données d'entraînement et de test ont la même distribution. Cette hypothèse signifie que les données d'entraînement et de test ont les mêmes caractéristiques, autrement dit que l'ensemble d'entraînement est représentatif du cadre d'application. En se plaçant dans cette hypothèse, il est raisonnable d'attendre une bonne généralisation aux données de test, sous réserve que le modèle atteigne une bonne performance pour les données d'entraînement et que celles-ci soient assez nombreuses, par rapport à la complexité de la tâche. Que faire lorsque l'hypothèse selon laquelle les données d'entraînement et de test ont les mêmes caractéristiques n'est pas vérifiée ? L'objet de ce chapitre est de présenter la littérature qui s'intéresse à cette question, en nous concentrant sur les systèmes de reconnaissance de la langue et sur l'impact du canal de transmission.

En apprentissage automatique, la notion de *domaine* est utilisée pour désigner une distribution des données. Nous introduisons d'abord cette notion dans le cadre de l'apprentissage supervisé. Ensuite, nous présentons plusieurs configurations de développement

et d'évaluation de systèmes, définies par les domaines utilisés lors de l'entraînement et de l'évaluation. Nous appelons *scénarios d'apprentissage* ces configurations. Puis, nous expliquons comment le canal de transmission peut être utilisé pour définir un domaine pour des données audios et discutons des hypothèses associées. Une fois ce cadre fixé, nous passons en revue les approches introduites dans la littérature pour limiter l'effet du domaine sur la performance des systèmes. Enfin, nous présentons les travaux qui visent à analyser l'impact du domaine sur un système.

## 3.1 La notion de domaine dans le cadre de l'apprentissage automatique

L'objectif de cette partie est de préciser la notion de domaine pour un système de reconnaissance de la langue. Nous montrons comment la notion de domaine émerge naturellement du paradigme d'apprentissage automatique.

### 3.1.1 Cadre de l'apprentissage automatique

Le champ de recherche de l'apprentissage automatique vise à développer des algorithmes capables de réaliser des tâches complexes avec la meilleure performance possible. Une tâche est définie par les entrées fournies à l'algorithme et par les sorties espérées. Ainsi pour un système de reconnaissance de la langue, l'entrée est un signal audio contenant de la parole et la sortie attendue un vecteur de scores pour chaque langue avec un score plus élevé pour la langue effectivement présente dans le signal d'entrée.

Nous utiliserons les notations suivantes. $\mathcal{X}$ est l'ensemble des signaux correspondant aux entrées du système. $\hat{\mathcal{Y}}$ est l'espace des prédictions qui correspond aux sorties du système. Un système $s$ est une fonction de l'ensemble des signaux d'entrée vers l'espace des prédictions.

$$s : \mathcal{X} \to \hat{\mathcal{Y}}$$

Le but de l'apprentissage automatique est la sélection d'un système pertinent pour réaliser cette tâche. En reconnaissance de la langue, les systèmes à l'état de l'art sont constitués de plusieurs modules qui font chacun l'objet d'une phase d'apprentissage indépendante (voir chapitre 2). Dans cette partie, nous nous limitons au cas où $s$ est constitué d'un seul module appris en une seule phase d'apprentissage, étant entendu que les principes que nous exposons peuvent être directement transposés à chaque module.

L'apprentissage automatique fonctionne en trois étapes. D'abord, on collecte des données d'entraînement représentatives de la tâche visée. L'ensemble fini des données d'entraînement sera noté $\mathcal{E}$ et son cardinal $N$. Ensuite, la connaissance *a priori* de la tâche est utilisée pour définir une famille $\mathcal{H}$ (famille d'hypothèses) de modèles pertinente pour réaliser la tâche. Enfin, l'algorithme d'apprentissage sélectionne un modèle performant parmi la famille $\mathcal{H}$ en s'appuyant sur les données d'entraînement.

En général, la famille $\mathcal{H}$ est une famille paramétrique de fonctions de $\mathcal{X}$ dans $\hat{\mathcal{Y}}$ :

$$\mathcal{H} = \{s_\theta | \theta \in \Theta\}$$

où $s_\theta$ est un modèle de la famille $\mathcal{H}$ indexé par le paramètre $\theta$ et où $\Theta$ est l'ensemble des paramètres admissibles qui définit la famille $\mathcal{H}$. Pour un réseau de neurones, la famille $\mathcal{H}$ est définie par l'architecture du réseau tandis que $\Theta$ correspond à l'ensemble des valeurs pouvant être prises par ses poids.

Sous quelle forme les données d'entraînement sont-elles disponibles lors de la phase d'apprentissage ? Un critère important est la présence d'annotations correspondant à la classe visée. Nous notons $\mathcal{Y}$ l'ensemble des étiquettes ou annotations.Pour un signal $x \in \mathcal{X}$, une étiquette $y \in \mathcal{Y}$ caractérise simplement la prédiction attendue du système. Pour un système de reconnaissance de la langue, l'étiquette est la langue associée au signal. Pour plus de clarté dans ce chapitre, nous noterons directement $\mathcal{Y}$ les étiquettes de langue, en passant sous silence la différence entre les étiquettes de langue $L$ et les classes du problème de classification $\mathcal{Y}$ évoquée dans le chapitre 2. Notons que l'espace des prédictions $\hat{\mathcal{Y}}$ n'est pas nécessairement identique à l'ensemble des étiquettes $\mathcal{Y}$. Si le modèle prédit directement la langue, alors on aura effectivement $\hat{\mathcal{Y}} = \mathcal{Y}$. En général, le système prédit un score pour chaque langue visée et on utilisera la notation $\hat{\mathcal{Y}}$. On distingue trois configurations en fonction de l'annotation des données d'entraînement. L'apprentissage supervisé correspond à la configuration où les données d'entraînement sont annotées. L'apprentissage non supervisé correspond à la configuration où les données d'entraînement ne sont pas annotées. L'apprentissage semi-supervisé est une situation intermédiaire où une partie des données d'entraînement seulement est annotée. Pour chacune de ces trois configurations, des algorithmes d'apprentissage spécifiques ont été introduits. Les modules des systèmes de reconnaissance de la langue sont généralement entraînés par apprentissage supervisé. Nous introduisons donc la notion de domaine dans ce cadre.

### 3.1.2 Apprentissage supervisé par minimisation du risque empirique

Dans une configuration d'apprentissage supervisé, les sorties attendues du système sont disponibles pour les données d'entraînement. Une stratégie naturelle apparaît donc : sélectionner le modèle de la famille $\mathcal{H}$ réalisant les meilleures prédictions pour les données d'entraînement dans l'espoir qu'il généralise bien aux données de test.

La mesure de la qualité du modèle est réalisée à l'aide d'une fonction de coût $l$ ($l$ pour *loss* en anglais). Les métriques présentées en section 2.2 pourraient être directement utilisées. En pratique, les réseaux de neurones sont entraînés par descente de gradient stochastique et on préfère donc des fonctions de coût différentiables. La fonction de coût attribue à une étiquette et à une prédiction réalisée par le système le coût de cette prédiction. Elle renvoie donc un coût élevé si la prédiction est erronée et faible si elle est exacte.

$$l : \mathcal{Y} \times \hat{\mathcal{Y}} \to \mathbb{R}_+$$

où

$$\mathcal{E} = \{(x_i, y_i)\}_{i=1}^{N} \in (\mathcal{X} \times \mathcal{Y})^N$$

Le coût $\mathcal{L}_\mathcal{E}(s_\theta)$ d'un modèle $s_\theta \in \mathcal{H}$ sur l'ensemble d'entraînement $\mathcal{E}$ est la moyenne de la fonction de coût pour les prédictions réalisées par le modèle sur cet ensemble. Il est

parfois appelé risque empirique.

$$\mathcal{L}_{\mathcal{E}}(s_\theta) = \frac{1}{N}\sum_{i=1}^{N} l(y_i, s_\theta(x_i))$$

L'algorithme d'entraînement appelé *minimisation du risque empirique* consiste à sélectionner un paramètre $\theta^\star \in \Theta$ minimisant ce coût [Vapnik, 1992]. On le notera parfois $\theta^\star_N$ pour rappeler que $|\mathcal{E}| = N$.

$$\theta^\star \in \underset{\theta\in\Theta}{argmin} \mathcal{L}_{\mathcal{E}}(s_\theta)$$

Sous quelles hypothèses un tel algorithme peut-il mener à une bonne performance sur l'ensemble des données de test ? Afin de pouvoir espérer apprendre une information utile sur les données de test à partir des données d'entraînement, il est indispensable de supposer un lien entre ces deux ensembles de données. L'hypothèse sous-jacente d'une large partie de la littérature est l'existence d'une distribution des données à partir de laquelle sont échantillonnées les données d'entraînement et de test. Une telle distribution est appelée *domaine*.

Un **domaine** $\mathcal{D}$ est une distribution sur $\mathcal{X} \times \mathcal{Y}$. On notera $p(x,y)$ la mesure associée, $p(x)$ et $p(y)$ les distributions marginales sur $\mathcal{X}$ et sur $\mathcal{Y}$, et $p(x|y)$ et $p(y|x)$ les distributions conditionnelles associées.

Dans ce cadre, faisons une hypothèse : les données d'entraînement sont indépendantes et identiquement distribuées selon la distribution $\mathcal{D}$. Sous cette hypothèse, des garanties théoriques peuvent être obtenues. Notons $\theta^\star_N$ un minimiseur du coût empirique pour un ensemble d'entraînement de taille $N$. Un minimiseur du coût empirique $\theta^\star_N$ converge vers un minimiseur du coût espéré $\mathcal{L}_{\mathcal{D}}(s_\theta)$ lorsque la taille $N$ de l'ensemble d'entraînement tend vers $+\infty$, où le coût espéré admet l'expression suivante [Vapnik, 2013].

$$\mathcal{L}_{\mathcal{D}}(s_\theta) = \underset{x,y\sim\mathcal{D}}{\mathbb{E}}\left[l(y, s_\theta(x))\right]$$

De plus, pour un modèle $s_\theta \in \mathcal{H}$, des bornes ont été démontrées sur l'écart entre le coût empirique et le coût espéré, en fonction de la complexité de la famille de fonctions $\mathcal{H}$, de la régularité de la fonction de coût $l$ et du cardinal $N$ de l'ensemble d'entraînement [Vapnik, 2013].

Si on fait également l'hypothèse que les données de test sont indépendantes et identiquement distribuées selon la distribution $\mathcal{D}$, alors l'espérance du coût sur l'ensemble de test n'est autre que le coût espéré.

Des garanties sur la généralisation d'un modèle entraîné par minimisation du risque empirique peuvent donc être obtenues en faisant l'hypothèse que les données d'entraînement et de test sont indépendantes et identiquement distribuées selon une même distribution appelée domaine $\mathcal{D}$. Une telle hypothèse signifie que les données partagent les mêmes caractéristiques. En suivant les définitions de [Arjovsky, 2020], cette hypothèse peut être justifiée dans le cas où le processus de génération des données est identique. C'est par

exemple le cas pour une application industrielle lorsque les données d'entraînement sont directement collectées dans le cadre où sera appliqué le système.

Pour la reconnaissance de la langue, deux des modules principaux du système sont des réseaux de neurones profonds : l'extracteur de *bottleneck features* et le réseau d'identification de la langue. Parfois le classifieur final est également un réseau de neurones. Ces réseaux de neurones ont un nombre de paramètres bien souvent supérieur de plusieurs ordres de grandeur au cardinal de l'ensemble d'entraînement. En conséquence, la complexité de la famille de fonctions $\mathcal{H}$ est très importante et nous ne sommes pas dans le cadre des bornes théoriques existantes.

En pratique, on applique plutôt des heuristiques qui s'inspirent de ce cadre théorique. La complexité de la famille de fonctions $\mathcal{H}$ est ajustée par des modifications de l'algorithme d'entraînement appelées méthodes de régularisation et discutées dans la section 2.4.4. Ces méthodes visent à atteindre une bonne généralisation aux données de test qui est un intermédiaire entre deux configurations insatisfaisantes : le sous-apprentissage et le sur-apprentissage [Goodfellow *et al.*, 2016]. Dans un cas de *sous-apprentissage*, la famille de fonctions $\mathcal{H}$ est de capacité trop faible (en pratique cela signifie que la régularisation est trop forte), le risque empirique minimum sur l'ensemble d'entraînement est donc élevé et la performance est faible. À l'inverse, dans un cas de *sur-apprentissage*, la famille de fonctions $\mathcal{H}$ est d'une trop grande complexité (la régularisation est trop faible), le risque empirique atteint sur l'ensemble d'entraînement est très faible mais cela ne se traduit pas par une bonne performance sur les données de test. Pour des réseaux de neurones profonds, ce cas pathologique est facile à atteindre en raison de la capacité importante des modèles.

### 3.1.3 Positionnement du travail

L'objet de notre travail est l'étude des systèmes de reconnaissance de la langue lorsqu'on s'affranchit de l'hypothèse selon laquelle les données d'entraînement et de test sont échantillonnées selon la même distribution. En particulier nous nous intéressons au cas où les différentes distributions correspondent à des canaux de transmission différents. Plusieurs raisons peuvent expliquer que les données d'entraînement et de test ne proviennent pas d'un même unique canal. D'abord la collecte de données du domaine visé dans le but de constituer un corpus d'entraînement peut être impossible ou simplement coûteuse. Nous pouvons également imaginer ne pas disposer de la capacité d'annoter des données de ce domaine et donc n'y accéder qu'à travers des données non annotées. Nous pouvons tout simplement ignorer les caractéristiques exactes du domaine sur lequel le modèle devra fonctionner et vouloir produire un modèle de classification le plus général possible. Dans un autre ordre d'idée, nous pouvons imaginer vouloir utiliser pour l'entraînement du modèle des données provenant d'un domaine différent du domaine cible, soit parce que nous ne disposons pas de suffisamment de données du domaine cible, soit parce que nous n'en disposons que pour un sous-ensemble des cas d'usage qui nous intéressent, par exemple seulement certaines langues.

Pour un canal de transmission $\mathcal{C}$, nous noterons $\mathcal{D}_\mathcal{C} \sim \mathcal{X} \times \mathcal{Y}$ le domaine associé correspondant à une distribution des signaux et étiquettes et $p_\mathcal{C}(x,y)$ la mesure associée.

## 3.2 Domaine défini par le canal de transmission

Dans cette partie, nous précisons la notion de domaine défini par le canal de transmission pour un signal audio. D'abord nous introduisons les hypothèses générales existant dans la littérature sur le lien entre deux domaines. Nous décrivons ensuite les différents facteurs de variabilité du signal de parole et définissons la notion de domaine associé au canal de transmission. Nous discutons ensuite les hypothèses associées à cette définition.

### 3.2.1 Lien entre deux domaines

Attardons nous sur les hypothèses qui peuvent être faites sur la relation entre les domaines. Notons donc deux domaines $\mathcal{D}_1$ et $\mathcal{D}_2$, correspondant à deux mesures $p_1(x,y)$ et $p_2(x,y)$ sur $\mathcal{X} \times \mathcal{Y}$. Nous utilisons ici les catégories de [Kull et Flach, 2014].

#### Aucune relation entre les domaines

Si aucune hypothèse n'est faite sur le lien entre $p_1$ et $p_2$ alors il est vain de tenter de comparer les deux distributions.

#### *Label shift* ou *prior probability shift*

$$p_1(y) \neq p_2(y) \quad \text{mais} \quad p_1(x|y) = p_2(x|y)$$

Dans ce cas, on suppose que pour une classe donnée la distribution des signaux est préservée par le changement de domaine. En revanche, les classes ne sont pas observées dans les mêmes proportions sur les deux domaines. À l'extrême, nous pouvons imaginer que les deux domaines correspondent à des signaux de mêmes caractéristiques mais que certaines classes ne sont observées que sur un domaine.

L'hypothèse de *label shift* est un cas relativement favorable dans le cadre d'un système de reconnaissance de la langue $L$ puisque celui-ci doit produire des rapports de vraisemblance $LR = \frac{p(x|y=L)}{p(x|y\neq L)}$. La valeur du rapport de vraisemblance est donc préservée entre les deux domaines. Seules sont modifiées les probabilités *a priori* des langues $p(y)$, par conséquent c'est le seuil appliqué aux rapports de vraisemblance qui devra être adapté en fonction des domaines.

#### *Covariate shift*

$$p_1(x) \neq p_2(x) \quad \text{mais} \quad p_1(y|x) = p_2(y|x)$$

Dans ce cas, les signaux présentent des distributions différentes mais la relation entre le signal $x$ et la classe $y$ est préservée. Comme c'est cette fonction $x \rightarrow y$ qui doit être estimée par le modèle, il semble possible d'apprendre un modèle robuste au changement de canal sur l'intersection du support des deux distributions. De nombreuses études sur la robustesse des systèmes se placent sous cette hypothèse.

***Concept shift***

$$p_1(x) = p_2(x) \quad \text{mais} \quad p_1(y|x) \neq p_2(y|x)$$

Dans ce cas, la distribution des signaux est la même sur les deux domaines. La relation entre le signal et la classe n'est cependant pas préservée. Il n'est donc pas possible d'apprendre un unique modèle qui prédirait la classe $y$ en fonction du signal $x$ et qui fonctionnerait sur les deux domaines.

**Combinaison de plusieurs hypothèses**

D'autres configurations sont imaginables, éventuellement en combinant les différences entre les domaines déjà exposées. Une hypothèse d'intérêt pratique est la combinaison des hypothèses de *label shift* et de *covariate shift*. Dans ce cas, les distributions des langues sont différentes ainsi que la distribution des signaux conditionnellement à la langue.

$$p_1(x|y) \neq p_2(x|y) \quad \text{et} \quad p_1(y) \neq p_2(y)$$

### 3.2.2 Les facteurs de variabilité du signal de parole

Précisons maintenant la notion de domaine pour un signal de parole. Dressons d'abord l'inventaire des différentes sources de variabilité du signal de parole. Nous appelons *facteur de variabilité* une caractéristique affectant le signal. Le terme *facteur de nuisance* est réservé au cadre d'une tâche et correspond aux *facteurs de variabilité* pouvant avoir un impact négatif sur la réalisation de la tâche. Ainsi la variabilité due au bruit est un facteur de nuisance pour de nombreux traitements de la parole. La variabilité due au locuteur peut être considérée comme un facteur de nuisance pour la reconnaissance de la langue tandis que ce ne sera évidemment pas le cas pour une tâche de reconnaissance du locuteur.

De nombreuses variables affectent le signal de parole, en premier lieu le *contenu linguistique* prononcé et la *durée* du signal. L'identité du *locuteur* a un impact sur les caractéristiques acoustiques du signal, notamment à travers le *genre* et l'*âge*, et également sur la langue utilisée, l'*accent*, et sur les phrases prononcées. Le *type de discours* (parole lue, conversation, discours, *interview*, *etc.*) a également un impact clair à la fois sur le contenu linguistique et directement sur le signal. D'autre facteurs de variabilité interviennent après la prononciation par le locuteur. L'*ambiance sonore* correspond aux perturbations du signal avant la captation : bruit additif, mélange avec un autre signal et réverbération. Le *dispositif de prise de son* caractérise les distorsions appliquées par les microphones et le *canal de transmission* les perturbations produites par le système de transmission. Ces facteurs de variabilité ont également un impact sur les caractéristiques de la parole prononcée puisqu'on peut considérer que, dans un environnement où la communication est rendue difficile par de fortes distorsions, le locuteur adapte son effort vocal et son articulation afin de maximiser son intelligibilité. Il s'agit de l'effet Lombard [Kadiri, 1998]. La notion de langue peut elle-même être exposée à des variabilités dépendant

éventuellement du locuteur telles que le *dialecte* ou l'*accent*. Lorsqu'un locuteur utilise plusieurs langues au sein d'un même enregistrement, on parle de *code switching*.

Un domaine peut être compris comme une distribution particulière des variables affectant le signal de parole, qui produit donc une distribution associée pour les signaux et langues. Un changement de domaine correspond à un changement de cette distribution. Notre étude s'intéresse tout particulièrement à un changement de canal de transmission.

### 3.2.3 Le canal de transmission

Listons quelques canaux de transmission communément utilisés pour la voix : l'enregistrement direct par un microphone, le réseau téléphonique, la téléphonie mobile, des émissions de radio ou de télévision, de la voix sur internet (*Voice over Internet Protocol*), des vidéos postées sur *Internet*, des canaux radios HF, VHF ou UHF.

L'utilisation d'un canal de transmission entraîne certaines distorsions au signal. D'abord, le canal impose un encodage particulier (*codec*) qui est parfois associé à une perte d'information. D'autre part, le canal a un effet convolutif en imposant un filtrage des différentes bandes de fréquence. Il est également associé à un bruit additif et convolutif. Certaines des distorsions sont dues aux caractéristiques du dispositif de transmission tandis que d'autres proviennent du milieu de transmission.

L'effet du canal de transmission sur un système de reconnaissance de la langue n'est plus à démontrer. C'est un des facteurs de variabilité qui a fait l'objet d'une thèse soutenue en 2011 [Verdet, 2011]. Récemment, la réduction de la performance due au changement de canal a été observée lors de la campagne d'évaluation *NIST LRE 2017* avec un écart entre les conversations téléphoniques et l'audio extrait de vidéos [Sadjadi *et al.*, 2018a].

### 3.2.4 Nature de la notion de domaine

Afin d'étudier l'impact du canal de transmission sur les performances de systèmes, nous définissons des domaines en fonction du canal de transmission. Une difficulté est le contrôle des autres facteurs de variabilité affectant le signal. Deux configurations sont possibles.

D'une part, le changement de domaine peut correspondre à une *intervention* [Arjovsky, 2020] sur le canal de transmission. Cela signifie que le canal de transmission est modifié tandis que les distributions des autres variables sont maintenues constantes (y compris le contenu linguistique). Le signal de parole n'est donc modifié qu'en fonction de sa dépendance directe au canal de transmission. C'est le cas de corpus artificiels, par exemple provenant d'augmentations de données ou comme le corpus *RATS* qui correspond aux mêmes signaux transmis à travers différents canaux réels.

D'autre part, le changement de domaine correspondant à un changement de canal peut être associé à un changement de distribution pour d'autres variables, par exemple le genre des locuteurs, le contenu linguistique ou l'ambiance sonore. C'est généralement le cas pour les corpus de données provenant d'applications réelles puisque chaque canal de transmission est associé à un contexte d'application. Des émissions de radio sont

associées à un enregistrement en studio avec une parole lue, tandis qu'un téléphone mobile peut être utilisé pour des conversations dans la rue (environnement bruité) ou dans une pièce (réverbération). Dans ce cas, le changement de canal doit être compris comme un changement de la distribution sur l'ensemble de ces variables.

Dans la littérature comme dans notre travail, ces deux configurations, *intervention* sur le canal de transmission et changement de distribution naturelle, ne sont pas distinguées. Cela se justifie dès lors que nous n'utilisons pas de modélisation de l'effet du canal mais nous limitons à l'information pouvant être extraite d'enregistrements de différents domaines, considérés comme des canaux. Notons toutefois que la configuration correspondant à une intervention sur le canal permet de mesurer précisément l'impact de celui-ci tandis que le changement de domaines naturels recouvre une notion plus vague de domaine correspondant au cadre de la collecte des données.

### 3.2.5 Hypothèses associées au changement de canal

À de très rares exceptions comme certains mots utilisés dans plusieurs langues, le contenu linguistique est associé à une langue unique. Dans notre travail, nous ferons toujours cette hypothèse. Par conséquent, dès lors que la parole est intelligible (ce qui doit toujours être le cas pour un canal de transmission), alors la langue est définie de façon univoque.

D'autre part, le canal de transmission entraîne des distorsions du signal, qui sont spécifiques à chaque canal. Le changement de domaine correspond donc à l'hypothèse de *covariate shift* : la distribution $p(x|y)$ varie en fonction du domaine.

Enfin, lorsque plusieurs domaines d'entraînement sont utilisés, si les stratégies d'échantillonnage diffèrent en fonction des domaines, alors nous pourrons être exposés à un *label shift* : les langues ne seront pas observées dans les mêmes proportions sur tous les domaines.

**Le problème du support** De même que les différentes langues sont associées à des supports différents pour les distributions $p(x|y)$, bien souvent les canaux de transmission $\mathcal{C}$ auront des supports disjoints pour les distributions $p_{\mathcal{C}}(x)$. Cela traduit simplement le fait que les distorsions appliquées au signal sont différentes pour différents canaux. Dès lors, sans faire d'autre hypothèse, la distribution $p_{\mathcal{C}_1}(x|y)$ pour un canal $\mathcal{C}_1$ ne porte aucune information sur la distribution $p_{\mathcal{C}_2}(x|y)$ pour un autre canal $\mathcal{C}_2$.

Pour espérer qu'un transfert soit possible entre différents domaines, nous devons décomposer le signal de parole en différentes variables latentes sur le modèle de [Arjovsky, 2020]. Autrement dit, nous devons supposer que l'information de langue peut être représentée par une variable latente dont les supports sont partagés pour différents canaux et qui autorise donc un transfert d'information. La difficulté dans ce cadre est l'estimation des variables latentes à partir du signal. La littérature qui s'intéresse au problème de la désintrication d'un signal en plusieurs variables latentes correspondant aux différents facteurs de variabilité est appelée *disentanglement* [Achille et Soatto, 2018].

## 3.3 Scénarios mettant en œuvre différents domaines

Nous avons défini la notion de domaine et les hypothèses sur la variation entre les domaines dans le cas où ils sont définis par le canal de transmission. Lors du développement d'un système, certains domaines seront utilisés pour l'entraînement et d'autres pour l'évaluation. Nous appelons *scénario* la caractérisation des domaines utilisés lors de l'entraînement et du test, ainsi que les hypothèses associées à ces domaines.

### 3.3.1 Adaptation de domaine

L'*adaptation de domaine* [Ben-David *et al.*, 2010] est un scénario faisant intervenir deux domaines appelés domaine source et domaine cible. Le but du scénario est la maximisation de la performance sur le domaine cible. L'ensemble d'entraînement est constitué principalement du domaine source. Une adaptation du modèle peut être réalisée lors de la phase d'entraînement à l'aide de données provenant du domaine cible. Si ces données sont annotées, on parle d'*adaptation de domaine supervisée*, les données du domaine cible seront alors disponibles en faible quantité. Si ces données ne sont pas annotées, on parle alors d'*adaptation de domaine non supervisée*.

La campagne d'évaluation *NIST LRE* 2017 [Sadjadi *et al.*, 2018b] a présenté un scénario d'adaptation de domaine supervisée. En plus des canaux de transmission habituels de cette série d'évaluation, téléphone et émissions de radio, le corpus de test contenait des enregistrements provenant de vidéos postées sur *Internet*. Ce domaine n'était accessible qu'à travers un petit ensemble de développement lors de l'entraînement.

La question principale soulevée par l'adaptation de domaine est celle du transfert d'information depuis le domaine source vers le domaine cible.

### 3.3.2 Apprentissage multi-domaines

L'*apprentissage multi-domaines* est un scénario où plusieurs domaines sont disponibles dans l'ensemble d'entraînement, le but étant de maximiser la performance sur l'ensemble de ces domaines [Joshi *et al.*, 2012]. Il est particulièrement intéressant en cas de *label shift*, lorsque certaines langues ne sont pas observées sur certains domaines lors de l'entraînement alors qu'elles sont présentes sur tous les domaines dans le corpus de test.

Les évaluations *NIST LRE* 2009, 2011 et 2015 [Zhao *et al.*, 2016] ont constitué des scénarios d'apprentissage multi-domaines. En effet deux canaux de transmission étaient utilisés à la fois pour l'entraînement et l'évaluation des systèmes : téléphone et *broadcast narrow band speech*. Les données d'entraînement pour chacune de ces évaluations présentent un *label shift* important avec certaines langues qui ne sont pas observées sur un des deux canaux.

### 3.3.3 Généralisation à un nouveau domaine

La *généralisation à un nouveau domaine* est un scénario difficile : le domaine de test ne fait pas partie des domaines utilisés lors de l'apprentissage [Arjovsky, 2020]. Plusieurs

domaines peuvent être utilisés lors de l'entraînement. Aucun algorithme n'a à ce jour une meilleure performance que la minimisation du risque empirique pour tous les domaines pour le scénario de généralisation à un nouveau domaine [Gulrajani et Lopez-Paz, 2020].

Nous pouvons cependant espérer améliorer la performance dans le cadre particulier du canal de transmission pour les systèmes de reconnaissance de la langue, puisqu'il s'agit d'une source de variabilité bien identifiée. La généralisation à un canal de transmission inconnu pour un système de reconnaissance de la langue a constitué une des tâches de la compétition *Oriental Language Recognition* lors des éditions 2019 [Tang *et al.*, 2019] et 2020 [Li *et al.*, 2020a].

## 3.4 Méthodes de limitation de l'impact du domaine sur la performance des systèmes

Nous évoquons maintenant les points de vue adoptés pour maximiser la performance d'un système en dépit de la présence d'un changement de domaine, en nous inspirant d'articles passant en revue les méthodes servant à l'adaptation de domaine [Wilson et Cook, 2020], l'apprentissage multi-domaines [Sun *et al.*, 2015] et la généralisation à un nouveau domaine [Gulrajani et Lopez-Paz, 2020, Zhou *et al.*, 2021, Wang *et al.*, 2021a]. Nous rapportons ici des méthodes appliquées aux systèmes reposant sur un réseau de neurones central, qu'ils soient appliqués à la reconnaissance de la langue ou la reconnaissance du locuteur. Notre travail porte sur la robustesse au canal de transmission. Certaines des approches présentées dans cette partie reposent sur une modélisation de l'effet du canal. D'autres en revanche sont simplement des méthodes d'apprentissage visant à prendre en compte plusieurs domaines, sans faire d'hypothèse sur la nature du domaine et donc potentiellement généralisable à d'autres facteurs de variabilité. Pour cette raison, nous rapportons des travaux portant sur la robustesse à de nombreux facteurs de variabilité : l'ambiance sonore et notamment le niveau de bruit, la réverbération, le type de discours, le genre du locuteur, la durée des enregistrements. Pour des systèmes de reconnaissance du locuteur, la langue utilisée est également parfois considérée comme un domaine. Deux points de vue dominent pour limiter l'impact du domaine sur la performance : la réduction de l'écart entre les domaines et le développement de modèles spécialisés pour chaque domaine.

### 3.4.1 Réduction de l'écart entre les domaines

Dans le cadre de l'adaptation de domaine, des bornes théoriques ont été démontrées pour caractériser la perte de performance entre deux domaines [Ben-David *et al.*, 2010, Muandet *et al.*, 2013, Mansour *et al.*, 2008]. Une revue récente de ces résultats théoriques est présentée dans [Redko *et al.*, 2020]. Ces résultats visent à proposer une majoration de la différence entre le coût empirique estimé sur le domaine source et le coût espéré sur le domaine cible. Ils reposent sur des hypothèses sur la complexité de la famille de fonctions utilisées pour apprendre le modèle et sur une mesure de l'écart entre les domaines. L'écart

entre les domaines est souvent une mesure de divergence entre les distributions associées à chaque domaine dans l'espace $\mathcal{X}$.

En reconnaissance de la langue, les familles de fonctions sont des réseaux de neurones profonds. Ils ne rentrent pas dans le cadre des résultats existants et rapportés dans [Redko *et al.*, 2020]. Ces résultats sont donc considérés comme des motivations à l'idée d'utiliser des représentations pour lesquelles l'écart entre les domaines est faible. Cette idée est particulièrement adaptée à l'hypothèse de *covariate shift* qui suppose que la différence entre les domaines réside entièrement dans la différence de distribution dans l'espace $\mathcal{X}$.

D'autre part, la minimisation de l'écart entre les domaines ne se limite pas à l'espace des signaux $\mathcal{X}$. Le système de reconnaissance de la langue peut être compris comme un extracteur de représentations de plus en plus abstraites, et l'écart entre les domaines peut donc être mesuré à tous les niveaux : dans l'espace des signaux, dans l'espace des *features acoustiques*, des *bottleneck features*, des *embeddings* et des scores.

Les approches qui visent à minimiser l'écart entre les domaines peuvent être regroupées en plusieurs catégories. D'abord les modèles utilisés peuvent simplement être choisis avec cet objectif. D'autre part, lors de l'inférence, les données de test peuvent être corrigées afin de les rendre plus similaires au domaine source. À l'inverse, les données des domaines d'entraînement peuvent être transformées avant l'entraînement du modèle dans le but de les rendre plus proches du domaine cible. Enfin, le critère de minimisation de l'écart entre les domaines peut être directement incorporé dans la sélection des paramètres des modules du système. Chacune des trois dernières approches peut être appliquée à plusieurs niveaux dans le système de reconnaissance de la langue.

Une transformation des données, d'entraînement ou de test, est parfois appelée une approche *feature based*, par opposition aux approches *model based* qui modifient les paramètres des modèles.

#### Sélection de représentations variant peu entre les domaines

La première stratégie pour minimiser l'écart entre les domaines est de sélectionner, parmi les systèmes proposés pour la reconnaissance de la langue, ceux qui sont le moins sujets à un écart entre les domaines. Cette sélection peut être opérée pour chacun des modules du système.

Certains *features* ont été introduits en raison de leur robustesse, par exemple des *features* basés sur des enveloppes temporelles dans des bandes de fréquence [Fernando *et al.*, 2018]. Les *bottleneck features* sont souvent considérés plus robustes aux conditions acoustiques [McLaren *et al.*, 2016a]. D'autre part les *features* peuvent être normalisés afin de minimiser leur variabilité [Padi *et al.*, 2019a], par exemple en fonction de la durée des segments [Shen *et al.*, 2018]. Une normalisation très répandue est la *CMVN* (*cepstral mean and variance normalisation*) [Molau *et al.*, 2003].

Une telle normalisation peut également avoir lieu dans l'extracteur d'*embeddings* par exemple en normalisant la moyenne sur la couche de *pooling* statistique d'un système *x-vector* [McLaren *et al.*, 2020, Shen *et al.*, 2020]. D'autre part il a été montré que les systèmes *x-vector* faisaient preuve d'une plus grande robustesse aux bruits additifs et à la réverbération que les systèmes *i-vectors* [Novotný *et al.*, 2018] tandis que certains travaux

indiquent que les réseaux de neurones récurrents sont plus sensibles au changement de canal que les systèmes *i-vectors* [Zazo *et al.*, 2016b]. L'analyse factorielle sur laquelle sont fondés les systèmes *i-vectors* repose sur l'idée de limiter l'impact des variabilités dues au canal et à la session sur les représentations [Matrouf *et al.*, 2011].

Dans le classifieur final, la *LDA* est souvent comprise comme une méthode de compensation de l'effet du canal [Misra *et al.*, 2016]. Un raffinement afin de compenser des facteurs de variabilité est la *Within Class Covarariance Correction* [Plchot *et al.*, 2017]. D'autres raffinements de la *LDA* ont été introduits pour limiter l'effet de la langue sur des représentations de reconnaissance du locuteur [Misra et Hansen, 2018]. La *Nuisance Attribute Projection* [Solomonoff *et al.*, 2007] est une autre projection des représentations qui permet de limiter l'impact de facteurs de variabilité, notamment le canal.

#### Correction des données de test

La correction des données de test afin de les rendre plus similaires aux données d'entraînement permet de réduire certaines variabilités, notamment le bruit. Lorsque ce traitement est effectué sur les *features* d'entrée du système, on parle de réhaussement de la parole ou de débruitage [Shon *et al.*, 2019]. Un tel traitement peut être réalisé par un réseau de neurones [Richardson *et al.*, 2016, Eskimez *et al.*, 2018, Frederiksen *et al.*, 2018] qui produit un signal propre à partir d'un signal bruité et a montré son intérêt pour un système basé sur des *i-vectors*.

Le débruitage peut aussi avoir lieu directement dans l'espace des *embeddings*, comme réalisé pour un système de reconnaissance du locuteur reposant sur des *x-vectors* [Mohammadamini *et al.*, 2020]. Un tel débruitage a également été réalisé dans un espace de *i-vectors* au moyen d'un auto-encodeur entraîné avec la *maximum mean discrepancy* [Lin *et al.*, 2018].

#### Transformation des données d'entraînement

L'approche symétrique à la correction des données de test est la transformation des données d'entraînement, dans le but d'entraîner le système dans des conditions plus proches des conditions d'inférence. Trois idées ont été utilisées.

L'augmentation de données consiste à appliquer des transformations au signal de parole afin d'augmenter la diversité des données observées par le système et *de facto* sa robustesse [Mccree *et al.*, 2016]. L'augmentation de données fait partie intégrante des recettes standards d'entraînement de réseau de neurones et a été discutée dans le chapitre précédent.

L'adapation *feature based* repose également sur une modification des données d'entraînement. Cependant au lieu de réaliser des transformations définies par des facteurs de variabilité connus du signal de parole, cette méthode vise à aligner explicitement les données du domaine source sur le domaine cible à l'aide d'une transformation simple. Des translations et multiplications matricielles dans le but de faire correspondre les deux premiers moments des distributions sont ainsi appliquées dans les espaces d'*embeddings* [Bousquet et Rouvier, 2019, Bousquet et Rouvier, 2020].

Enfin, une partie de la littérature d'adaptation de domaine vise à reproduire la distribution du domaine cible par un rééchantillonage de données du domaine source. Cela n'est cependant pas possible lorsque les supports des deux distributions sont disjoints comme c'est le cas pour le canal. Cependant lors d'un apprentissage multi-domaines, il est possible d'équilibrer les différentes conditions lors de l'entraînement dans le but de limiter l'impact de la différence entre les domaines. Cela a été réalisé lors du calcul de l'entropie croisée [Trong *et al.*, 2018] et pour entraîner un classifieur Gaussien [McLaren *et al.*, 2018b].

#### Minimisation de l'écart entre les domaines *model based*

La minimisation de l'écart entre les domaines peut être directement utilisée comme objectif lors de l'entraînement du système. On parle alors de construction de représentations *invariantes*. Des *i-vectors* invariants au domaine ont été introduits [Rahman *et al.*, 2018].

Ce type d'approche a pris un essor avec les systèmes basés sur des réseaux de neurones profonds. En effet, pour un réseau de neurones, une mesure de l'écart entre les domaines peut être directement ajoutée à la fonction de coût du système et minimisée au cours de l'entraînement en même temps que la tâche initialement utilisée. Les méthodes diffèrent en fonction de la mesure utilisée pour l'écart entre les domaines. Pour rendre l'optimisation possible par une descente de gradient stochastique, il convient de choisir une fonction différentiable et pouvant être estimée à partir d'un échantillon de taille faible (le *minibatch*). Citons quelques mesures de l'écart entre les domaines couramment utilisées. *Deep CORAL* (*correlation alignement*) [Sun et Saenko, 2016] est une mesure de la distance entre les matrices de covariance des deux domaines, la *maximum mean discrepancy* (ou *kernel norm*) [Long *et al.*, 2015, Lin *et al.*, 2018] mesure la distance entre les moyennes des deux distributions dans un espace de Hilbert à noyau reproduisant. Des méthodes fondées sur le transport optimal réalisent un appariement entre les éléments des deux domaines [Damodaran *et al.*, 2018]. L'apprentissage antagoniste (*adversarial*) [Ganin *et al.*, 2016] repose sur l'utilisation d'un discriminateur qui est entraîné à reconnaître le domaine et dont la fonction de coût est utilisée pour mesurer l'écart entre les domaines. L'apprentissage antagoniste a été largement utilisé en reconnaissance du locuteur pour augmenter la robustesse à la langue [Rohdin *et al.*, 2019, Bhattacharya *et al.*, 2019b, Xia *et al.*, 2019] et aux conditions acoustiques [Bhattacharya *et al.*, 2019a, Meng *et al.*, 2019]. En reconnaissance de la langue, il a permis de réduire l'impact du canal pour des langues slaves [Abdullah *et al.*, 2020]. Le transport optimal a été appliqué lors de la compétition *Oriental Language Recognition 2020* [Lu *et al.*, 2021]. Remarquons que ces approches n'utilisent généralement pas d'annotation en langue pour la mesure de l'écart entre les domaines. Elles peuvent donc être appliquées à des scénarios d'adaptation de domaine non supervisée.

Dans [Cai *et al.*, 2020a], la réduction de l'écart entre les canaux dans l'espace des *embeddings* est envisagée d'un autre point de vue. Au lieu de mesurer la différence entre les canaux avec des données non annotées de canaux différents, la fonction de coût *within-sample similarity loss* caractérise le canal comme une information ne variant pas au

cours d'un enregistrement par opposition au contenu linguistique. Une régularisation est ajoutée à la fonction de coût de l'extracteur d'*embeddings* lors de son entraînement afin de supprimer cette information. Cette méthode est intéressante puisqu'elle ne requiert pas d'utiliser des données provenant de différents canaux.

L'entraînement antagoniste a également été appliqué à l'extracteur de *bottleneck features* [Peng *et al.*, 2019].

### 3.4.2 Un traitement dépendant du domaine

Plutôt que tenter de réduire l'écart entre les domaines, il est possible de spécialiser le système pour améliorer sa performance sur un ou plusieurs domaines cibles spécifiques. Dans ce cas, il peut être nécessaire d'utiliser l'information de domaine lors de l'inférence.

#### Adaptation spécifique au domaine cible

L'approche la plus directe consiste à entraîner un système différent pour chaque condition [Mccree *et al.*, 2018]. Lorsque les données d'entraînement ne sont pas disponibles en quantité pour tous les domaines, il faut se résoudre à partager l'entraînement de certains modules du système entre les domaines.

Une adaptation spécifique d'une partie du système au domaine cible peut être réalisée. Pour un réseau de neurones, cela peut correspondre à une étape appelée *finetuning* (raffinement) [Lopez *et al.*, 2018] : quelques époques de descente de gradient sont réalisées sur des données du domaine cible après avoir pré-entraîné le modèle sur un autre domaine. Cela a été réalisé pour adapter un système de reconnaissance du locuteur au type de discours (neutre ou stress physique) par un *finetuning* de l'extracteur d'*embeddings* ainsi qu'un entraînement du classifieur final sur le domaine cible [Zhang *et al.*, 2018].

De nombreuses méthodes d'adaptation du classifieur final au domaine cible ont été proposées [Verdet *et al.*, 2010, Plchot *et al.*, 2018, Bahmaninezhad *et al.*, 2021]. Le développement de telles méthodes était l'objectif de la compétition *2015 NIST i-vector machine learning challenge* [Tong *et al.*, 2016] qui fournissait le système d'extraction de *i-vectors* aux participants en leur proposant de se concentrer sur le développement d'un classifieur final. Pour la reconnaissance du locuteur, on compte des méthodes d'adaptation non supervisée de la *PLDA* comme *CORAL+* [Lee *et al.*, 2019], ainsi que des approches supervisées [Richardson *et al.*, 2016, Bousquet et Rouvier, 2020]. En reconnaissance de la langue, l'adaptation basée sur le *maximum a posteriori* d'un classifieur Gaussien domine [Richardson *et al.*, 2018, Lopez *et al.*, 2018].

L'adaptation la plus aisée peut être réalisée au niveau des modules de fusion des sous-systèmes et de calibration. Dans ce cas, il est courant de faire intervenir des informations caractérisant le domaine dans le calcul du score, comme la durée [Mccree *et al.*, 2016, McLaren *et al.*, 2018b] ou certains facteurs liés à la qualité du signal [Nautsch *et al.*, 2016].

**Prédiction du domaine lors de l'inférence**

Si un système spécifique a été développé pour chaque domaine, il est primordial d'aiguiller chaque enregistrement de test vers le bon système lors de l'inférence. Pour cela, il faut être capable de prédire le domaine avec une fiabilité acceptable. Il a été observé qu'il était possible d'améliorer la performance de reconnaissance de la langue grâce à une prédiction du canal de transmission pour un corpus constitué de deux canaux [Verdet, 2011].

### 3.4.3 Autres approches

D'autres méthodes ont été proposées afin de prendre en compte plusieurs domaines lors de l'entraînement d'un système, mais n'ont pas été appliquées à la reconnaissance de la langue. Parmi elles mentionnons l'approche *Invariant Risk Minimization* [Arjovsky *et al.*, 2020]. Elle ne consiste pas à produire des représentations qui ne varient pas entre les domaines mais des représentations telles que le classifieur optimal qui opère sur ces représentations soit le même pour tous les domaines. L'invariance est donc comprise comme une invariance de la distribution $p(y|x)$ et non pas comme celle de la distribution $p(x)$.

### 3.4.4 Comparaison des approches

Une comparaison systématique de l'ensemble de ces approches n'a pas été réalisée. Ce travail est malaisé puisque la performance des approches dépend étroitement des types de modèles sur lesquels elles sont appliquées. En règle générale, il apparaît que la meilleure performance peut être obtenue par la combinaison de plusieurs de ces approches.

Une question importante est celle du module du système sur lequel l'adaptation au domaine est la plus pertinente. Une telle étude a été réalisée pour l'adaptation au canal de transmission d'un système de reconnaissance de la langue reposant sur des *i-vectors* [McLaren *et al.*, 2016a]. Après un entraînement sur le canal téléphonique, l'ensemble des modules finaux ont été entraînés successivement sur le domaine cible, depuis la calibration jusqu'à l'*UBM*. Le réentraînement de chaque module apporte un gain, mais celui-ci est nettement plus important pour l'extracteur de *i-vectors* et le classifieur final que pour l'*UBM* et le module de calibration. Dans le même ordre d'idées, une comparaison a été faite pour un système de reconnaissance du locuteur entre une adaptation *feature based* des *embeddings* et une adaptation *model based* du classifieur, et ce à la fois pour des *i-vectors* et *x-vectors* [Alam *et al.*, 2018]. Il apparaît qu'il est préférable d'adapter uniquement les *embeddings* pour les *i-vectors*, et à la fois les *embeddings* et le classifieur pour les *x-vectors*.

### 3.4.5 Utilisation de données non annotées

Certaines méthodes permettent d'améliorer la robustesse des systèmes en utilisant des données non annotées. Elles sont notamment utiles dans un scénario d'adaptation de domaine non supervisée. Certaines des approches évoquées précédemment (celles basées

sur la minimisation *model based* d'une mesure de l'écart entre les domaines) peuvent être employées avec des données non annotées. Deux autres approches existent : l'apprentissage de représentations avec une tâche non supervisée et l'auto-annotation.

#### Apprentissage de représentations avec une tâche non supervisée

Des données non annotées peuvent être valorisées en apprenant des représentations avec une tâche non supervisée ou auto-supervisée. Par exemple un coût de reconstruction pour les données non annotées a été utilisé lors de l'entraînement d'un classifieur final constitué d'un réseau de neurones [Ben-Reuven et Goldberger, 2016].

Ces dernières années, des tâches auto-supervisées, c'est-à-dire ne nécessitant pas d'annotation ont été utilisées pour apprendre des *features* utiles pour d'autres tâches. L'approche [Pascual *et al.*, 2019] utilise plusieurs objectifs auto-supervisés, comme le calcul de *features* acoustiques ou la prédiction de trames en fonction de leur contexte. Les résultats les plus impressionnants pour une telle méthode ont été obtenus en reconnaissance de la parole [Baevski *et al.*, 2020] à l'aide d'une tâche *contrastive* qui consiste à prédire un vecteur latent, appris par le système, en fonction de son contexte. En reconnaissance de la parole, il a été montré qu'une grande partie de la perte de performance due à un changement de domaine pouvait être compensée par l'ajout de données non annotées du domaine cible lors de la phase d'entraînement auto-supervisée de l'extracteur de *features* [Hsu *et al.*, 2021].

En reconnaissance de la langue, une telle tâche *contrastive* a été utilisée pour entraîner l'extracteur de *features* sans toutefois atteindre la performance de *bottleneck features* [Ling *et al.*, 2020].

#### Auto-annotation

L'auto-annotation (*self-labeling*) est une méthode permettant d'incorporer des données non annotées à l'ensemble d'entraînement, notamment dans un scénario d'adaptation de domaine non supervisée. Elle consiste à utiliser un modèle préalablement entraîné pour annoter de nouvelles données, *a priori* provenant d'un domaine différent. Ces nouvelles données sont ensuite ajoutées à l'ensemble d'entraînement et un nouveau modèle est entraîné [Nercessian *et al.*, 2016]. Cette approche a été utilisée dans le cadre d'un système *i-vector*. L'auto-annotation suppose que, même si la performance du modèle sur le nouveau domaine est faible, les prédictions contiennent une information pertinente et peuvent être utilisées pour raffiner le modèle en incorporant de nouveaux domaines dans l'ensemble d'entraînement. C'est donc une approche adaptée pour un écart assez faible entre les domaines.

## 3.5 Analyse de systèmes

Au-delà de la conception de systèmes robustes, la question du canal doit être envisagée du point de vue de l'analyse d'un système existant.

Un grand intérêt est porté sur la nature de l'information extraite par les modèles et particulièrement par celle qui est portée par les *embeddings*. Ainsi, les auteurs de [Raj *et al.*, 2019] ont montré que les *x-vectors* utilisés dans un système de reconnaissance du locuteur contiennent des informations caractérisant le canal, la méthode d'augmentation appliquée aux enregistrements, la durée des segments et même la transcription. Dans la même logique, il a été montré que l'information de canal est préservée tout au long d'un extracteur d'*embeddings* caractérisant le dialecte [Chowdhury *et al.*, 2020]. Ces éléments semblent indiquer que des facteurs de nuisance tels que le canal ne sont pas éliminés par l'extracteur d'*embeddings*. Une prédiction indépendante des facteurs de nuisance doit être réalisée en compensant leur effet.

Une meilleure compréhension de l'information portée par les représentations devrait permettre de mieux anticiper le comportement des systèmes sur de nouveaux domaines. Ainsi des travaux en traitement du texte ont permis de prédire la performance sur un nouveau domaine en fonction de mesures de l'écart avec le domaine d'entraînement dans un espace de représentation [Elsahar et Gallé, 2019].

En traitement de la parole, des premiers travaux apparaissent. Ainsi il a été proposé d'analyser l'importance de certains facteurs de qualité (niveau de bruit, identité du locuteur) sur des systèmes de détection d'attaques contre un système de reconnaissance du locuteur (*anti-spoofing*) à l'aide de mesures de divergence entre des *features* calculés sur différents corpus [Chettri *et al.*, 2021].

## 3.6 Positionnement de ce travail

À l'aune de cette revue des travaux existants, nous avons concentré notre étude sur une approche : la minimisation de l'écart entre les domaines dans l'espace des *embeddings*, par la production de représentations invariantes (chapitre 4). Les méthodes d'augmentation de la robustesse dépendant étroitement de la nature des modèles utilisés, nous avons travaillé sur des méthodes applicables à des réseaux de neurones profonds, puisque nous faisons le constat que la plupart des modules du système sont progressivement remplacés par de tels modèles. De plus, la dynamique d'intégration des différents modules actuellement à l'œuvre devrait permettre d'appliquer cette approche à l'ensemble du système, correspondant alors à un réseau de neurones *end-to-end*.

Les travaux de cette thèse ont débuté en 2018 et font suite à l'affirmation des *x-vectors* comme modèle d'*embeddings* plus performant que les *i-vectors*. Par conséquent les méthodes de limitation de l'impact de facteurs de variabilité principalement utilisées étaient celles qui s'appliquaient au classifieur final, qui avaient été développées pour les systèmes *i-vectors*. Cependant, des travaux suggéraient que l'adaptation des *embeddings* était une approche plus performante qu'un travail sur le classifieur final. Notre étude sur l'adaptation de domaine non supervisée s'inscrit dans une série de travaux en 2019 et 2020 qui ont ajouté une fonction de coût à l'extracteur de *x-vectors* afin d'augmenter leur invariance à certains facteurs de variabilité.

Nous avons ensuite montré que l'augmentation de l'invariance des *embeddings* était une approche générale qui pouvait s'appliquer à d'autres scénarios que l'adaptation de

domaine non supervisée. Nous avons discuté les modalités de sa mise en œuvre pour un apprentissage multi-domaines et la généralisation à un domaine inconnu. Nous avons confirmé l'efficacité de cette approche pour la généralisation à un canal inconnu lors de la compétition *Oriental Language Recognition 2020*.

D'abord dans le but de comprendre l'effet de nos méthodes sur les représentations puis avec l'objectif d'anticiper la variation de la performance sur un nouveau domaine, nous mesurons l'impact de facteurs de variabilité sur les représentations produites par nos systèmes (chapitre 5). Une telle analyse est relativement nouvelle en reconnaissance de la langue.

# 4

# Robustesse d'un système de reconnaissance de la langue au canal de transmission

L'objectif de notre travail est l'augmentation de la robustesse d'un système de reconnaissance de la langue au canal de transmission. D'abord, nous rappelons, à partir d'exemples, la perte de performance due au changement de canal de systèmes standards. Nous montrons que cette perte de performance est principalement due à l'absence de données des canaux d'intérêt dans le corpus d'entraînement, ou de leur observation en proportion insuffisante. Nous explorons une approche pour réduire cet écart consistant à augmenter l'invariance des représentations au canal de transmission. Comme exposé dans la section 3.4.1, la minimisation de l'écart entre les domaines peut être réalisée par la régularisation du réseau de neurones d'identification de la langue. La régularisation est obtenue par l'addition d'une fonction de coût visant à un rapprochement des distributions des représentations correspondant à différents domaines. Nous étudions deux types de fonctions de régularisation : celles qui mesurent l'écart entre deux distributions et celles qui visent à un rapprochement des représentations des échantillons correspondant à la même langue, dites de *metric learning*. Nous mettons en évidence l'intérêt de ces approches dans trois scénarios d'apprentissage mettant en œuvre un changement de canal : l'adaptation de domaine non supervisée, l'apprentissage multi-domaines et la généralisation à un domaine inconnu.

## 4.1 Impact du changement de canal de transmission

Le canal de transmission entraîne des distorsions du signal, sans toutefois modifier le contenu linguistique. Nous attendons d'un système de reconnaissance de la langue qu'il se concentre sur le contenu linguistique en faisant abstraction des variations dues au canal. Nous montrons cependant que ce n'est pas le cas. Pour différentes recettes standards d'entraînement de systèmes, le changement de canal induit des variations importantes de

TABLE 4.1 – Architecture du système *GMM*.

| Détection de la parole, extraction de *features* acoustiques fournis en entrée d'un *GMM* [Torres-Carrasquillo *et al.*, 2002] produisant une log-vraisemblance par langue, calibration des log-vraisemblances. | | |
|---|---|---|
| Module | Modèle | Hyper-paramètres |
| détection d'activité vocale | énergie du signal | - |
| *features* acoustiques | *SDC-MFCC* | $\Delta_F = 20ms$, $\delta_F = 10ms$, 16 filtres *Mel* |
| augmentation de données | aucune | - |
| *bottleneck features* | non utilisés | - |
| modélisation du segment | *GMM* | un *GMM* par langue avec 4096 composantes et des matrices de covariance diagonales |
| classifieur final | non utilisé | utilisation directe des log-vraisemblances produites par les *GMM* |
| calibration | régression logistique | outil *Focal* [Brümmer, 2007] |

performance.

Dans cette première partie, nous rendons compte de l'effet du changement de canal de transmission sur les performances d'un système de reconnaissance de la langue. Plusieurs expériences sont menées avec différents systèmes et différents corpus dans le but de mettre en évidence un comportement partagé par toutes les recettes standards d'entraînement de système. Les systèmes de reconnaissance de la langue subissent une perte de performance dès qu'ils sont testés sur un canal de transmission non ou peu observé lors de l'entraînement.

### 4.1.1 Test sur un canal inconnu

Notre première série d'expériences vise à évaluer la performance sur un canal inconnu, avec les données de la compétition *Oriental Language Recognition 2020*. Nous comparons un système classique et un système à l'état de l'art et faisons varier la proportion de données de canaux inconnus dans le corpus d'entraînement.

#### Description des systèmes

Nous comparons deux systèmes. Le premier, décrit dans le tableau 4.1, est un mélange de Gaussiennes (*GMM*) utilisant des *features* de type *SDC-MFCC*, système de référence du début des années 2000 pour les systèmes acoustiques. Le second, décrit dans le tableau 4.2, repose sur un réseau de neurones utilisant des *bottleneck features* multilingues.

TABLE 4.2 – Architecture du système basé sur un réseau de neurones d'identification de la langue.

| Détection de la parole, extraction de *features* acoustiques, puis de *bottleneck features*, fournis en entrée d'un réseau d'identification de la langue produisant une probabilité *a posteriori* par langue, calibration des log-vraisemblances. | | |
|---|---|---|
| Module | Modèle | Hyper-paramètres |
| détection d'activité vocale | énergie du signal | - |
| *features* acoustiques | spectrogrammes en échelle *Mel* | $\Delta_F = 25ms$, $\delta_F = 10ms$, 64 filtres *Mel*, *CMVN* |
| augmentations de données | quatre augmentations | *specAugment* [Park *et al.*, 2019], ajout de bruit blanc, et *babble noise*, filtres passe-bande aléatoires |
| *bottleneck features* | réseau *Conformer* [Gulati *et al.*, 2020] | entraîné avec la *CTC-loss* pour dix langues avec les augmentations de données, recette d'entraînement détaillée dans le chapitre 6 |
| modélisation du segment | *TDNN* standard [Snyder *et al.*, 2018] | entraîné pour 16 langues avec l'entropie croisée, les augmentations de données et la stratégie *stochastic weight averaging* |
| classifieur final | non utilisé | utilisation directe des probabilités *a posteriori* produites par le réseau |
| calibration | régression logistique | outil *Focal* [Brümmer, 2007] |

TABLE 4.3 – Performance en terme de $C_{avg} \times 100$ des systèmes en fonction du corpus d'entraînement et pour deux ensembles de test du corpus de l'évaluation *Oriental Language Recognition 2020*.

| Système | | $C_{avg} \times 100$ | |
|---|---|---|---|
| | | Corpus de test | |
| Architecture | Corpus d'entraînement | *mobile* | *canaux inconnus* |
| Réseau de neurones | *mobile* | 5,60 | 10,30 |
| Réseau de neurones | *tous canaux* | 4,69 | **7,26** |
| *GMM* | *tous canaux* | **2,53** | 10,82 |

### Corpus utilisés

Nous utilisons ici les données des compétitions *Oriental Language Recognition*. En 2019 et 2020, la compétition proposait une tâche de reconnaissance de la langue sur un canal inconnu, la majorité des données d'entraînement provenant de téléphones mobiles [Tang *et al.*, 2019, Li *et al.*, 2020a]. Notre évaluation est réalisée pour les six langues de la première tâche de la compétition *OLR 2020* : cantonais, coréen, indonésien, japonais, russe et vietnamien. Nous constituons deux ensembles de test, qui correspondent à la phase de développement des systèmes appelée *conception du système final* en annexe A.3 :

— *test mobile* : sous-ensemble du corpus *AP18-test* et constitué uniquement de données de téléphone mobile

— *test mobile et canaux inconnus* : sous-ensemble des corpus *AP19-dev-tâche2* et *AP19-test-tâche1*, constitué de canaux inconnus ainsi que de téléphone mobile.

Nous entraînons les systèmes avec deux corpus différents :

— *entraînement mobile* (corpus d'entraînement de la phase appelée *sélection d'un modèle robuste* en annexe A.3) : sous-ensemble des corpus *AP16-OL7*, *AP17-OL3* et *AP17-test*.

— *entraînement tous canaux* (corpus d'entraînement de la phase appelée *conception du système final* en annexe A.3) : sous-ensemble des corpus *AP16-OL7*, *AP17-OL3*, *AP17-test*, *AP18-test*, *AP19-dev-tâche2*, *AP19-test* et *AP20- entraînement*.

Pour chaque système, la calibration est effectuée sur un corpus de validation ayant les mêmes caractéristiques que le corpus d'entraînement. La composition détaillée des corpus est présentée en annexe A.3. Notons qu'il n'y a pas de recouvrement entre les corpus d'entraînement et de test.

### Analyse des performances

Le tableau 4.3 présente les performances de trois systèmes. La performance est évaluée en terme de $C_{avg}$, métrique de référence de la communauté de reconnaissance de la langue, avec le point de fonctionnement ($C_{FA} = C_{FR} = 1$, $P_{cible} = 0{,}5$). Nous utilisons toujours ce point de fonctionnement dans ce document.

En première observation, la performance est toujours dégradée sur les canaux inconnus par rapport au téléphone mobile. Ensuite, bien que le système *GMM* atteigne la meilleure performance sur le canal mobile ($C_{avg} \times 100 = 2{,}53$), il subit la plus forte dégradation de performance pour le canal inconnu (10,82) et celle-ci est très supérieure à celle subie par un système basé sur un réseau de neurones (10,30 au lieu de 5,60). Cette observation conforte notre travail sur les systèmes basés sur les réseaux de neurones pour améliorer la robustesse au canal. Enfin, pour un système basé sur un réseau de neurones, l'utilisation d'un corpus d'entraînement plus important améliore nettement les performances sur les deux corpus de test. L'amélioration très significative sur l'ensemble contenant des enregistrements de canaux inconnus (7,26) peut être expliquée par la présence de canaux inconnus dans le second corpus d'entraînement.

En résumé, les systèmes à l'état de l'art subissent une dégradation de performance due au changement de canal. Cette dégradation est moins importante que pour un système *GMM*. La portée de cette comparaison est cependant limitée puisque nous n'avons pas mis en œuvre les méthodes établies d'augmentation de la robustesse de systèmes *GMM*. Pour le système reposant sur un réseau de neurones, la perte de performance est réduite lorsqu'on ajoute des canaux différents dans le corpus d'entraînement.

### 4.1.2 Entraînement sur plusieurs canaux

Nous nous concentrons maintenant sur les systèmes dont le module central est un réseau de neurones d'identification de la langue, tels qu'ils sont décrits dans le chapitre 2. Pour de tels systèmes, l'écart de performance entre les canaux est-il imputable à une difficulté intrinsèque de la tâche, plus grande pour des canaux présentant des distorsions importantes, ou bien à la non observation de canaux au cours de l'entraînement ? Pour répondre à cette question, nous réalisons des expériences symétriques en entraînant et testant des systèmes sur les neuf canaux téléphonique et radios du corpus *RATS*.

#### Description du système

Nous utilisons un système *x-vector* fondé sur l'extraction d'un *embedding* (tableau 4.5). Ce système ne comprend pas de module de détection d'activité vocale, car le corpus *RATS* contient des annotations en activité vocale, que nous utilisons. De même, nous ne nous préoccupons pas de la calibration et utilisons une mesure de performance y étant insensible : la moyenne des taux d'égale erreur par langue ($EER_{avg}$).

Ce système *x-vector* repose sur l'architecture standard *TDNN* [Snyder *et al.*, 2018] décrite dans le tableau 4.4. Les cinq premières couches sont des convolutions à une dimension avec un contexte, éventuellement en sautant certaines trames. La couche de *pooling* statistique extrait moyenne et écart-type. Toutes les non-linéarités sont des *rectified linear units* ($ReLU$). Des normalisations par *batch* sont appliquées sur les couches opérant un traitement par trame. Les *embeddings* sont les activations de la couche *segment6* tandis que des scores d'identification de la langue sont produits par la couche *softmax*.

TABLE 4.4 – Architecture du réseau *TDNN* [Snyder *et al.*, 2018]. $d(= d_F$ ou $d_B)$ est la dimension des *features* d'entrée, $T(= T_F$ ou $T_B)$ le nombre de trames du segment d'entrée et $n$ le nombre de langues reconnues par le modèle.

| Couche | Contexte de la couche | Contexte cumulé | entrée × sortie |
|---|---|---|---|
| trame1 | $[t-2,t+2]$ | 5 | $5d \times 512$ |
| trame2 | $\{t-2,t,t+2\}$ | 9 | $1536 \times 512$ |
| trame3 | $\{t-3,t,t+3\}$ | 15 | $1536 \times 512$ |
| trame4 | $\{t\}$ | 15 | $512 \times 512$ |
| trame5 | $\{t\}$ | 15 | $512 \times 1500$ |
| *pooling* statistique | $[0,T)$ | T | $1500T \times 3000$ |
| segment6 | $\{0\}$ | T | $3000 \times 512$ |
| segment7 | $\{0\}$ | T | $512 \times 512$ |
| *softmax* | $\{0\}$ | T | $512 \times n$ |

**Corpus utilisés**

Nous utilisons le corpus *RATS* qui contient neuf canaux de transmission. Des conversations enregistrées sur un canal téléphonique (nommé *src*) ont été transmises à travers huit canaux radio : A, B, C, F, G (UHF), E (VHF), D et H (HF). Comme les mêmes conversations sont présentes sur tous les canaux, nous n'utilisons que la moitié du corpus pour réaliser des entraînements, dans le but de nous ménager la possibilité de réaliser des adaptations avec la partie des données non observée. Un schéma représentant le principe de la découpe est exposé en figure 4.1. Nous utilisons les corpus *RATS SAD* et *RATS KWS* qui comptent cinq langues : anglais, arabe, farsi, ourdou et pachto. Nous constituons des corpus d'entraînement, de validation et de test avec des segments de trois secondes. Une description précise des corpus est donnée en annexe A.2.

**Performance pour un entraînement par canal et pour un entraînement multi-canaux**

Nous entraînons un système pour chaque canal et le testons sur tous les canaux. De plus, nous entraînons un modèle supplémentaire avec un ensemble d'entraînement contenant tous les canaux (mais de taille comparable aux autres ensembles d'entraînement en prenant un sous-ensemble des enregistrements). Nous évaluons la performance en terme de moyenne des taux d'égale erreur ($EER_{avg}$), dans le but d'éliminer une partie de l'erreur de calibration et de nous concentrer sur l'impact du canal d'entraînement du réseau de neurones d'identification de la langue. En mesurant la performance en terme de $C_{avg}$, on observerait un écart de performance encore plus grand entre les canaux, puisqu'une erreur de calibration s'ajouterait aux erreurs venant de la capacité de discrimination des scores.

Les résultats, rapportés dans le tableau 4.6 montrent clairement la dégradation des performances lorsque les systèmes sont utilisés sur des canaux non vus lors de l'entraînement. Des performances acceptables sont atteintes sur la diagonale lorsque le système

TABLE 4.5 – Architecture du système *x-vector*

| Extraction de *bottleneck features* multilingues, calculés à partir de spectrogrammes en échelle *Mel*, puis production d'un *x-vector* par le *TDNN* entraîné pour l'identification de la langue, fourni en entrée d'un *SVM* qui prédit la langue. | | |
|---|---|---|
| Module | Modèle | Hyper-paramètres |
| détection d'activité vocale | annotation manuelle | - |
| *features* acoustiques | log-spectrogrammes en échelle *Mel* | $\Delta_F = 25ms$, $\delta_F = 10ms$, prétraitement décrit dans [Silnova *et al.*, 2018] |
| augmentations de données | non utilisés | - |
| *bottleneck features* | *stacked bottleneck features* | modèle multilingue produisant des *features* de dimension 80. Nous utilisons le modèle de *BUT / Phonexia* [Silnova *et al.*, 2018] entraîné à prédire des triphones pour 17 langues du corpus *Babel*. |
| modélisation du segment | *TDNN* standard [Snyder *et al.*, 2018] (tableau 4.4) | entraîné pour l'identification de la langue avec l'entropie croisée |
| classifieur final | *SVM* | *LDA*, centrage, blanchiment des *embeddings* et prédiction de scores par un *SVM* |
| calibration | non utilisée | - |

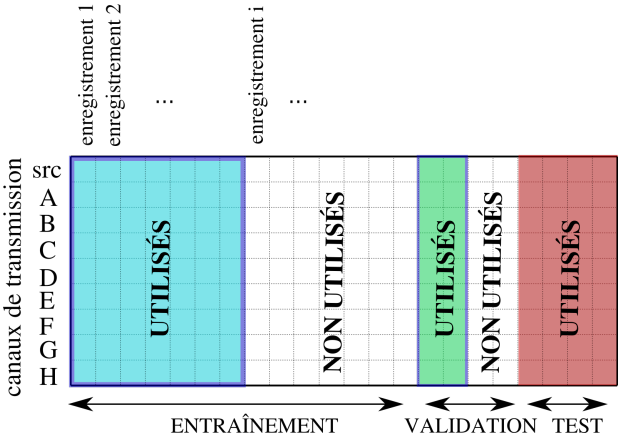



FIGURE 4.1 – Découpe du corpus *RATS* pour l'évaluation de la performance sur un canal inconnu. Seule la moitié du corpus d'entraînement est utilisée. Les blocs colorés sont utilisés pour l'entraînement, la validation et le test. Les blocs blancs ne sont pas utilisés pour cette série d'expériences.

est testé sur le canal pour lequel il a été entraîné ($EER_{avg}$ compris entre 6% et 15%). En dehors de la diagonale, la dégradation de performance est très importante. Notons l'exception du canal G, présentant de faibles distorsions et qui semble proche du canal téléphonique. À noter qu'un système entraîné sur tous les canaux atteint une bonne performance sur chacun d'entre eux (entre 5% et 12%).

De cette étude, nous tirons deux conclusions. D'une part, un système entraîné sur un seul canal subit une forte dégradation de performance sur un autre canal, y compris pour différents canaux radios. D'autre part, une bonne performance peut être atteinte sur plusieurs canaux cibles si ceux-ci sont représentés dans les données d'entraînement.

## 4.2 Régularisation de la fonction de coût du réseau d'identification de la langue

Forts du constat de la non robustesse au canal des systèmes actuels, notre travail consiste à améliorer les recettes d'entraînement afin de réduire l'écart de performance entre les canaux. Nous explorons une méthode pour augmenter la robustesse des systèmes au changement de canal de transmission.

### 4.2.1 Principe

Les réseaux de neurones sont constitués de couches successives. Les activations de chaque couche peuvent être considérées comme des représentations du signal d'entrée, contenant une partie de l'information portée par celui-ci. Les *embeddings* extraits d'un réseau (*bottleneck features* ou *x-vector*) sont des exemples de telles représentations. La nature de l'information sélectionnée par les activations est dirigée par l'algorithme d'ap-

TABLE 4.6 – Performance du système *x-vector* entraîné pour cinq langues sur le corpus *RATS* en fonction des canaux d'entraînement et de test. $EER_{avg}$ (%).

| | Canal de test | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Canal d'entraînement | src | A | B | C | D | E | F | G | H |
| src (téléphone) | **6** | 50 | 42 | 34 | 39 | 48 | 45 | **17** | 43 |
| A (UHF) | 42 | **15** | 40 | 43 | 39 | 49 | 47 | 48 | 49 |
| B (UHF) | 45 | 32 | **12** | 53 | 39 | 48 | 52 | 53 | 44 |
| C (UHF) | 45 | 35 | 40 | **13** | 39 | 43 | 50 | 46 | 34 |
| D (HF) | 38 | 35 | 38 | 47 | **7** | 47 | 45 | 41 | 49 |
| E (VHF) | 37 | 58 | 51 | 42 | 52 | **14** | 47 | 39 | 36 |
| F (UHF) | 38 | 44 | 42 | 42 | 40 | 45 | **13** | 39 | 42 |
| G (UHF) | **11** | 46 | 44 | 33 | 34 | 44 | 32 | **9** | 36 |
| H (HF) | 52 | 42 | 52 | 35 | 47 | 43 | 49 | 45 | **14** |
| src - A - B - C - D - E - F - G - H | **5** | **10** | **10** | **10** | **7** | **12** | **13** | **7** | **12** |

prentissage du réseau de neurones. Or l'entraînement du réseau est une descente de gradient stochastique dont l'objectif est défini par la fonction de coût.

Il est donc naturel, dans le but d'imposer une propriété nouvelle au réseau, de tenter d'encoder cette propriété directement dans la fonction de coût. Dans ce travail, nous nous intéressons à la robustesse au canal de transmission.

Comment dès lors encoder la propriété de robustesse au canal dans la fonction de coût ? Un domaine induit une distribution sur les activations des couches abstraites du réseau.Une manière d'assurer que le modèle atteigne des performances similaires sur tous les domaines est d'imposer que les distributions des activations pour chaque domaine dans l'espace des représentations soient similaires.

### 4.2.2 Régularisation de la fonction de coût du réseau d'identification de la langue

Nous modifions donc la fonction de coût du réseau d'identification de la langue afin de produire des représentations à la fois utiles à la tâche de classification et invariantes entre les domaines. Cela se traduit par la fonction objectif suivante :

$$L = \mathcal{L}_{classification} + \lambda \mathcal{L}_{\text{régularisation}}$$

où $\lambda$ est un hyper-paramètre correspondant au poids de l'objectif d'invariance. $\mathcal{L}_{classification}$ est une fonction de coût de classification, classique dans les recettes d'entraînement de systèmes. $\mathcal{L}_{\text{régularisation}}$ est la fonction de coût chargée d'imposer une proximité des représentations provenant des différents domaines.

De nombreux travaux ont appliqué ce principe pour développer des systèmes robustes à certaines variabilités. L'exemple le plus célèbre est probablement le *Domain Adversarial Neural Network* [Ganin *et al.*, 2016]. Nous explorons deux types de fonctions de coût visant à augmenter l'invariance : les mesures de l'écart entre les les distributions associées aux différents domaines et les fonctions de type *metric learning*.

### 4.2.3 Fonctions mesurant l'écart entre les domaines

Ces fonctions de coût sont directement inspirées de la borne de [Ben-David *et al.*, 2007]. Elles consistent à estimer la distribution des représentations de chaque domaine, et à mesurer l'écart entre ces distributions. N'importe quelle mesure de divergence entre deux mesures peut donc être utilisée en théorie.

Néanmoins, pour être applicable dans la pratique pour l'entraînement d'un réseau de neurones, deux propriétés sont nécessaires. La mesure de divergence doit pouvoir être estimée raisonnablement à partir d'un *mini-batch* de taille finie. D'autre part elle doit être différentiable par rapport aux éléments du *mini-batch* afin de permettre une retro-propagation du gradient.

Puisqu'elles mesurent l'écart entre les domaines, ces fonctions de coût exigent de disposer d'annotations en canal, ou du moins d'être capable d'estimer le canal avec un degré de certitude acceptable. En revanche, elles n'utilisent pas l'information de langue et peuvent donc être mises en œuvre dans le cadre d'un scénario d'apprentissage non supervisé ou semi-supervisé.

Notons que l'apprentissage *adversarial* est un cas particulier de divergence puisque, sous l'hypothèse d'un discriminateur optimal, il correspond à une divergence de Jensen-Shannon [Goodfellow *et al.*, 2020]. Dans notre travail, nous avons réalisé des expériences avec trois fonctions de coût mesurant l'écart entre les domaines : la distance entre les moyennes, *correlation alignment* [Sun et Saenko, 2016] et *maximum mean discrepancy* [Long *et al.*, 2015, Lin *et al.*, 2018].

Dans cette section, nous notons $\mathcal{D}_1$ et $\mathcal{D}_2$ les distributions induites par les deux domaines dans l'espace de représentation où sont appliquées les fonctions de régularisation. Ce seront dans nos expériences les activations d'une couche du réseau de neurones régularisé.

#### Distance entre les moyennes

Une façon élémentaire de caractériser la différence entre deux distributions est d'utiliser la distance entre leurs moyennes.

$$\mathcal{L}_{moyennes} = ||\mu_1 - \mu_2||_2^2 \quad \text{avec } \mu_1 = \mathop{\mathbb{E}}_{x_1 \sim \mathcal{D}_1}[x_1] \quad \text{et } \mu_2 = \mathop{\mathbb{E}}_{x_2 \sim \mathcal{D}_2}[x_2]$$

#### *Correlation Alignment*

En allant plus loin, on peut utiliser la distance entre les matrices de covariance (norme de Frobenius). Cette fonction de coût est appelée *Correlation Alignment* (*CORAL*) et elle a donné de bons résultats en traitement de l'image et du texte [Sun et Saenko, 2016].

$$\mathcal{L}_{CORAL} = ||\Sigma_1 - \Sigma_2||_F^2 \quad \text{où } \Sigma_1 = \mathop{\mathbb{E}}_{x_1 \sim \mathcal{D}_1}[(x_1 - \mu_1)(x_1 - \mu_1)^T] \quad \text{et } \Sigma_2 = \mathop{\mathbb{E}}_{x_2 \sim \mathcal{D}_2}[(x_2 - \mu_2)(x_2 - \mu_2)^T]$$

***Maximum Mean Discrepancy***

La *maximum mean discrepancy* (*MMD*) est une généralisation de la distance entre les moyennes en prenant le *supremum* de la distance entre les moyennes pour un espace de fonctions $\mathcal{H}$.

$$\mathcal{L}_{MMD} = \sup_{h \in \mathcal{H}, |h|=1} | \mathop{\mathbb{E}}_{x_1 \sim \mathcal{D}_1}[h(x_1)] - \mathop{\mathbb{E}}_{x_2 \sim \mathcal{D}_2}[h(x_2)] |^2$$

Si $\mathcal{H}$ est un espace de Hilbert à noyau reproduisant $k$ alors ce *supremum* peut être directement estimé à partir d'échantillons de la valeur du noyau.

$$\mathcal{L}_{MMD} = \mathop{\mathbb{E}}_{x_1, x_1' \sim \mathcal{D}_1}[k(x_1, x_1')] + \mathop{\mathbb{E}}_{x_2, x_2' \sim \mathcal{D}_2}[k(x_2, x_2')] - 2 \mathop{\mathbb{E}}_{x_1 \sim \mathcal{D}_1, x_2 \sim \mathcal{D}_2}[k(x_1, x_2)]$$

De plus, cette fonction de coût est différentiable et estimable à partir d'un échantillon fini [Peyré *et al.*, 2019]. Le calcul peut être parallélisé efficacement sur *GPU* grâce à la librairie *keops* [Feydy *et al.*, 2019].

Dans notre travail, nous avons utilisé deux noyaux. Le premier, appelé noyau énergie, est l'opposé de la distance $L_2$ entre deux échantillons.

$$k_{\text{energie}}(x_1, x_2) = -||x_1 - x_2||_2$$

Le second, le noyau gaussien, dépend d'un hyper-paramètre, la variance $\sigma^2$, qui représente l'échelle des variations prises en compte par le noyau.

$$k_{\text{gaussien}}(x_1, x_2) = exp(-\frac{||x_1 - x_2||_2^2}{\sigma^2})$$

Enfin, notons qu'en utilisant un noyau linéaire on retrouve à une constante multiplicative près la fonction de coût de distance entre les moyennes $\mathcal{L}_{moyennes}$.

$$k_{\text{linéaire}}(x_1, x_2) = x_1^T x_2$$

### 4.2.4 Fonctions de coût de type *metric learning*

Une fonction de coût de type *metric learning* a pour but d'assurer que les proximités géométriques entre représentations des échantillons soient cohérentes avec la tâche de classification. Plus précisément, elle opère sur des paires d'échantillons et vise à imposer que deux échantillons correspondant à la même classe soient plus proches l'un de l'autre que deux échantillons de classes différentes. C'est donc une approche supervisée qui a besoin des étiquettes de langues. En revanche, l'information de domaine n'est pas utilisée, ce qui en fait une approche de choix lorsque nous voulons réduire l'impact d'un facteur de variabilité mal connu. Dans cette partie, nous notons $\mathcal{D}$ la distribution des données d'entraînement dans l'espace des représentations.

Nous avons étudié deux fonctions : la *triplet loss* [Hoffer et Ailon, 2015] et la *n-pair loss* [Sohn, 2016, Kulkarni *et al.*, 2020].

***Triplet loss***

C'est la fonction de *metric learning* la plus couramment utilisée.

$$\mathcal{L}_{triplet} = \underset{(x,y),(x^+,y^+),(x^-,y^-)\sim\mathcal{D}|y^+=y,y^-\neq y}{\mathbb{E}}[max(0,||x-x^+||_2^2 - ||x-x^-||_2^2 + m)]$$

$x$ est appelé l'ancre, $x^+$ un échantillon positif de la même classe que $x$ et $x^-$ un échantillon négatif d'une autre classe, $y$, $y^+$ et $y^-$ étant les étiquettes de langues associées. $m \in \mathbb{R}_+$ est un hyper-paramètre appelé la marge.

La *triplet loss* a été utilisée pour entraîner le classifieur final dans [Mingote *et al.*, 2019].

***N-pair loss***

En notant $n$ le nombre de classes du problème d'identification de la langue, la *n-pair loss* est une version de la *triplet loss* avec $n-1$ exemples négatifs correspondant à chacune des autres classes. Plus précisément, si l'échantillon $x$ associé à l'étiquette $y$ est l'ancre, alors on échantillonne $n-1$ échantillons négatifs $x_1, \dots x_{n-1}$ associés aux étiquettes $y_1, \dots y_{n-1}$ et l'ensemble des étiquettes $\{y,y_1^-,\dots,y_{n-1}^-\}$ recouvre les $n$ classes $\{1,\dots,n\}$.

$$\mathcal{L}_{n-pair} = \underset{(x,y),(x^+,y^+),(x_1^-,y_1^-),\dots(x_{n-1}^-,y_{n-1}^-)\sim\mathcal{D}|y^+=y,\{y,y_1^-,\dots,y_{n-1}^-\}=\{1,\dots,n\}}{\mathbb{E}}[ln(1+\sum_{i=1}^{n-1} exp(x^T x_i^- - x^T x^+)]$$

**Stratégie d'échantillonnage**

Afin d'améliorer encore la robustesse, différentes stratégies ont été imaginées pour échantillonner les exemples positifs et négatifs. Une large part de la littérature propose des approches regroupées sous le terme de *negative mining* [Mingote *et al.*, 2019] qui visent à sélectionner des exemples maximisant la fonction de coût au cours de l'apprentissage. Si une autre information est disponible, elle peut être utilisée lors de l'échantillonnage. Dans le cas de la robustesse au canal, on pourrait pour maximiser l'invariance échantillonner des exemples positifs d'un canal différent de l'ancre et des exemples négatifs de même canal que l'ancre. Nous n'avons pas évalué cette idée expérimentalement.

### 4.2.5 Alignement entre données parallèles

Le comportement le plus souhaitable pour les représentations est qu'elles ne varient pas en fonction du domaine. Si on dispose d'un corpus aligné $\mathcal{D}_{1,2}$ avec des enregistrements du même signal sur les deux domaines, alors on peut minimiser directement la distance entre les deux.

$$\mathcal{L}_{parallele} = \underset{(x_1,x_2)\sim\mathcal{D}_{1,2}}{\mathbb{E}}[||x_1 - x_2||_2^2]$$

En général, on ne dispose pas d'un tel corpus aligné. Une telle propriété ne peut être obtenue que de façon artificielle, c'est le cas du corpus *RATS*. Nous utiliserons cependant cette fonction de coût afin de réaliser des expériences préliminaires pour sélectionner les hyper-paramètres de méthodes d'adaptation de domaine. De plus, une telle fonction de coût peut être utilisée dans le cas où un domaine est produit artificiellement à partir d'un autre. C'est le cas pour l'augmentation de données, omniprésente dans les recettes d'entraînement de systèmes.

### 4.2.6 Démarche expérimentale

Dans la suite de ce chapitre, nous montrons comment la régularisation de la fonction de coût du réseau d'identification de la langue permet d'augmenter la robustesse au changement de canal. Nous étudions trois scénarios d'apprentissage qui mettent en jeu un changement de canal : l'adaptation de domaine non supervisée, l'apprentissage multi-domaines et la généralisation à un nouveau domaine. Pour chacun de ces scénarios, nous discutons l'emploi optimal des différentes fonctions de coût proposées.

## 4.3 Adaptation de domaine non supervisée

### 4.3.1 Description du scénario

L'adaptation de domaine non supervisée est un scénario d'apprentissage où le but est la maximisation de la performance sur un domaine cible (*target* en anglais) $\mathcal{D}_\mathcal{T}$. On suppose disposer de deux corpus de données : des données annotées d'un domaine source $\mathcal{D}_\mathcal{S}$ différent du domaine cible et de données non annotées du domaine cible $\mathcal{D}_\mathcal{T}$. Il s'agit donc d'une tâche d'apprentissage semi-supervisé (car le corpus d'apprentissage mêle données annotées et non annotées) mais d'une adaptation non supervisée (car les données du domaine cible ne sont pas annotées). Ce scénario a un grand intérêt pratique. En effet, il semble raisonnable de collecter des données correspondant au contexte d'une application afin de développer un système. L'annotation de ces données doit cependant être réalisée par un annotateur humain pouvant distinguer les langues d'intérêt, ce qui se révèle très coûteux en temps. Il est donc souhaitable de limiter au maximum l'annotation de données du domaine cible. À l'inverse, de grandes bases de données annotées sont disponibles pour des canaux largement étudiés par l'industrie et la communauté académique, comme le canal téléphonique. Une stratégie d'adaptation de domaine non supervisée tente de tirer parti de ces bases de données en ajoutant au corpus d'entraînement des données non annotées correspondant au canal d'utilisation.

### 4.3.2 Mise en œuvre de la régularisation

Pour un tel scénario, les approches de type *metric learning* ne sont pas applicables car elles supposent un apprentissage supervisé. Nous nous limitons donc aux fonctions de coût caractérisant l'écart entre les domaines. Lors de l'entraînement de réseaux d'identification

de la langue, nous minimiserons la fonction de coût suivante.

$$\mathcal{L} = \mathcal{L}_{classification}(\mathcal{D}_{\mathcal{S}}) + \lambda \mathcal{L}_{\text{régularisation}}(\mathcal{D}_{\mathcal{S}}^{|\mathcal{X}}, \mathcal{D}_{\mathcal{T}}^{|\mathcal{X}})$$

où les notations $\mathcal{D}_{\mathcal{S}}^{|\mathcal{X}}$ et $\mathcal{D}_{\mathcal{T}}^{|\mathcal{X}}$ indiquent que la fonction de régularisation $\mathcal{L}_{\text{régularisation}}$ utilise uniquement des données non annotées des domaines $\mathcal{D}_{\mathcal{S}}$ et $\mathcal{D}_{\mathcal{T}}$.

Dans cette partie, nous sélectionnons la couche sur laquelle appliquer la régularisation, nous comparons les différentes fonctions de régularisation proposées, nous analysons l'effet de l'hyper-paramètre $\lambda$ associé à la régularisation et nous comparons cette approche à d'autres méthodes d'adaptation de domaine.

### 4.3.3 Contexte expérimental

Pour l'adaptation de domaine non supervisée, nous utilisons le corpus *RATS*, décrit dans la sous-partie 4.1.2, avec des segments de trois secondes. Au début de ce travail, en 2018, nous n'avions pas encore acquis le corpus *RATS*. Nous avons donc réalisé nos premières expériences avec un sous-ensemble du corpus *RATS* appelé *OpenSAD 15*. Ce corpus de taille moins importante ne contient que six canaux radios (il ne contient pas les canaux A et C) et nous n'utilisons que quatre langues. Nous excluons le farsi en raison de sa faible représentation dans le corpus.

Comme dans la sous-partie 4.1.2, nous découpons les corpus d'entraînement en deux parties afin de ne pas utiliser les mêmes enregistrements sur les canaux source et cible. Un schéma représentant le principe de la découpe est exposé en figure 4.2.

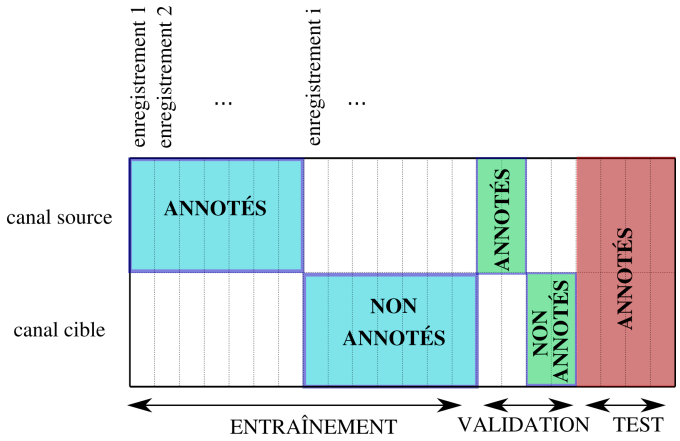



FIGURE 4.2 – Découpe des corpus *RATS* et *OpenSAD 15* pour les expériences d'adaptation de domaine non supervisée. Les blocs colorés sont utilisés pour l'entraînement, la validation et le test. Pour l'entraînement et la validation, chaque enregistrement n'est utilisé que sur un seul canal, soit avec une annotation de langue sur le canal source, soit sans annotation de langue sur le canal cible. Les blocs blancs ne sont pas utilisés.

Nous réalisons des expériences avec deux systèmes : un système *CNN* basé sur un réseau convolutionnel traitant des *MFCC* (tableau 4.8), et le système *x-vector* présenté dans la section 4.1.2 (tableau 4.5).

L'architecture du réseau convolutionnel utilisé par le systèmes *CNN* est présentée dans le tableau 4.7. Elle contient trois couches de convolution selon l'axe temporel avec un *max pooling*, suivies d'une couche de *pooling* statistique et de deux couches pleinement connectées. Les non linéarités sont des *rectified linear units* ($ReLU$), excepté sur la couche de sortie où il s'agit d'un *softmax*.

TABLE 4.7 – Architecture du réseau de neurones convolutionnel.
$n$ est le nombre de langues reconnues par le modèle. Dans nos expériences avec le corpus *OpenSAD 15*, $n = 4$.

| **Convolutions 1D selon l'axe temporel** | | |
|---|---|---|
| nom de la couche | taille du noyau / du *max pooling* | nombre de filtres |
| *conv1* | 5 / 2 | 1024 |
| *conv2* | 5 / 2 | 1024 |
| *conv3* | 5 / 2 | 128 |
| **Agrégation statistique des moyennes et écarts-types (*pooling*)** | | |
| | dimension de sortie : | $2 \times 128 = 256$ |
| **Couches connectées** | | |
| nom de la couche | dimension | |
| *fc1* | $256 \times 128$ | |
| *fc2* | $128 \times n$ | |

Le systèmes *CNN* (tableau 4.8) est entraîné sur le corpus *OpenSAD 15* et le système *x-vector* (tableau 4.5) sur le corpus *RATS*. Toujours comme dans la section 4.1.2, nous nous limitons à une évaluation des systèmes en terme de moyenne des taux d'égale erreur ($EER_{avg}$), dans le but de nous concentrer sur la capacité de discrimination entre les langues des systèmes, sans aborder le problème de la calibration.

Les premières expériences ont été réalisées avec le système *CNN*. Pour ces expériences préliminaires, le domaine source est le canal G et le domaine cible le canal D. Cela nous permet de sélectionner des hyper-paramètres en fonction de la performance d'identification de la langue obtenue sur le domaine cible. Dans un second temps, nous appliquons ces hyper-paramètres à un cadre réaliste d'adaptation où le canal téléphonique est le domaine source et où les canaux radios sont les domaines cibles. Le canal G a été sélectionné car c'est le canal radio présentant la dégradation de performance la plus faible par rapport au canal téléphonique, tandis que le canal D a été choisi comme canal HF, supposé particulièrement difficile.

### 4.3.4 Sélection de la couche régularisée

Sur quelle couche appliquer la fonction de régularisation qui impose l'invariance des représentations ? Il est certain que nous souhaitons une invariance au canal des prédictions du modèle, ce qui correspond à la couche de sortie. Cependant, une telle invariance pourrait être atteinte en imposant une invariance à une couche intermédiaire du réseau, ce qui donnerait une liberté plus grande aux couches de classification. Le réseau d'iden-

TABLE 4.8 – Architecture du système *CNN*.

| Prédiction directe de scores d'identification de la langue par un réseau convolutionnel prenant en entrée des *MFCC*. | | |
|---|---|---|
| Module | Modèle | Hyper-paramètres |
| détection d'activité vocale | annotation manuelle | - |
| *features* acoustiques | *MFCC* | $\Delta_F = 25ms$, $\delta_F = 10ms$, 13 coefficients |
| augmentations de données | aucune | - |
| *bottleneck features* | non utilisés | - |
| modélisation du segment | réseau de neurones convolutionnel (tableau 4.7) | entraîné pour l'identification de la langue avec l'entropie croisée |
| classifieur final | non utilisé | utilisation directe des probabilités *a posteriori* produites par le réseau |
| calibration | non utilisée | - |

tification de la langue ayant pour but de produire une représentation pour l'ensemble du segment, nous nous sommes limités à appliquer la régularisation sur les couches hautes, après la couche de *pooling* statistique.

Afin d'évaluer sur quelle couche l'invariance est la plus souhaitable, nous utilisons la régularisation exploitant l'alignement de données parallèles, ce qui est rendu possible par la nature du corpus *RATS*. Cette fonction de coût, bien qu'inapplicable dans un cadre d'adaptation de domaine réaliste, rend compte exactement de la propriété d'invariance souhaitée. C'est donc une fonction de coût idéale dont nous nous servons pour déterminer la couche la plus pertinente pour appliquer la régularisation, en espérant que la conclusion ne dépende pas de la fonction de régularisation utilisée.

Pour le système *CNN* et sur le corpus *OpenSAD 15*, en utilisant le canal G comme domaine source et le canal D comme domaine cible, nous procédons à plusieurs entraînements du réseau convolutionnel d'identification de la langue avec la régularisation exploitant l'alignement parallèle des données. La fonction de coût minimisée est la suivante :

$$\mathcal{L} = \mathcal{L}_{classification}(R(\mathcal{D}_{\mathcal{S}})) + \lambda \mathcal{L}_{parallele}(R_e(\mathcal{D}_{\mathcal{S},\mathcal{T}}^{|\mathcal{X}\times\mathcal{X}}))$$

où $\mathcal{D}_{\mathcal{S},\mathcal{T}}^{|\mathcal{X}\times\mathcal{X}}$ correspond à la distribution jointe d'enregistrements parallèles dans les espaces de *features* des domaines $\mathcal{D}_{\mathcal{S}}$ et $\mathcal{D}_{\mathcal{T}}$. $R(x)$ correspond à la sortie du réseau d'identification de la langue $R$ pour l'entrée $x$. $R_e(x)$ correpond aux activations d'une couche intermédiaire.

Nous utilisons trois couches intermédiaires dans nos expériences, correspondant aux couches traitant l'ensemble du segment : *pooling*, *fc1* et *fc2*. Dans le cas où la couche concernée est *fc2*, alors $softmax \circ R_e = R$. L'architecture complète du réseau est dé-

TABLE 4.9 – Performance en terme de moyenne des taux d'égale erreur (%) du système *CNN* sur le corpus *OpenSAD 15* pour quatre langues et des segments de trois secondes. L'adaptation est réalisée entre les canaux G et D. Trois types de méthodes d'apprentissage sont comparées : la régularisation par alignement parallèle, la régularisation sur la couche de sortie et l'apprentissage supervisé. Les noms des couches sont définis dans le tableau 4.7.

| | | $EER_{avg}$ (%) sur le canal de test | |
|---|---|---|---|
| | | source (G) | cible (D) |
| Régularisation par alignement parallèle | Couche | | |
| | *pooling* | **11** | 29 |
| | *fc1* | **11** | 21 |
| | *fc2* | 13 | **17** |
| Régularisation sur la couche *fc2* | Fonction | | |
| | distance entre les moyennes | 12 | 39 |
| | CORAL | **11** | 10 |
| | MMD | **11** | **9** |
| Apprentissage supervisé | Canaux d'entraînement | | |
| | G | **14** | 48 |
| | D | 53 | 15 |
| | src, B, D ,E, F, G, H | 15 | **12** |

crite dans le tableau 4.7. Dans nos expériences, $\mathcal{L}_{classification}$ est l'entropie croisée. Pour chaque choix de la couche $R_e$, nous sélectionnons l'hyper-paramètre $\lambda$ en fonction de la performance d'identification de la langue sur un ensemble de validation du domaine cible.

En première observation des résultats du tableau 4.9, la régularisation par alignement parallèle permet d'améliorer la performance de reconnaissance de la langue sur le domaine cible pour les trois couches étudiées ($EER_{avg}$ de 29%, 21% et 17% au lieu de 48%). Cet effet est plus grand sur la couche de sortie *fc2* (17%). Dans la suite, nous appliquons donc les fonctions de régularisation sur la couche de sortie.

D'autre part, nous observons un effet moins attendu. La régularisation du réseau a permis d'améliorer également la performance sur le domaine source, notamment lorsque la régularisation est appliquée sur les couches intermédiaires (11% au lieu de 14%). Dans la suite de notre travail, nous serons attentifs à ce phénomène. La construction de représentations invariantes n'est pas seulement utile à la généralisation à un nouveau domaine, elle peut aussi améliorer les représentations sur le domaine source et donc mener à une meilleure classification.

### 4.3.5 Comparaison des fonctions de coût

Nous comparons maintenant trois fonctions de coût non supervisées afin de réduire l'écart entre les domaines : la distance entre les moyennes, *CORAL* et la *MMD* avec

noyau Gaussien. La fonction de coût minimisée est la suivante.

$$\mathcal{L} = \mathcal{L}_{classification}(R(\mathcal{D}_{\mathcal{S}})) + \lambda \mathcal{L}_{\text{régularisation}}(R_e(\mathcal{D}_S^{|\mathcal{X}}), R_e(\mathcal{D}_T^{|\mathcal{X}}))$$

où $R_e$ correspond aux activations de la couche de sortie du réseau $R$ (avant le *softmax*), notée *fc2* dans le tableau 4.7.

D'abord, l'hyper-paramètre $\lambda$ ainsi que la variance $\sigma^2$ pour le noyau Gaussien sont sélectionnés pour une adaptation du canal G vers le canal D, pour le système *CNN*. Les performances de reconnaissance de la langue sont rapportées dans le tableau 4.9. Les trois régularisations améliorent la performance sur le domaine cible, ainsi que sur le domaine source. Les régularisations *CORAL* et *MMD* sont les plus efficaces avec des taux d'égale erreur moyens de 10% et 9% sur le domaine cible.

Une fois les hyper-paramètres sélectionnés pour une adaptation entre les canaux G et D, nous utilisons ces mêmes hyper-paramètres pour une adaptation du canal téléphonique (*src*) vers chacun des canaux radios. Les résultats sont rapportés dans le tableau 4.10.

TABLE 4.10 – Performance en terme de moyenne des taux d'égale erreur de différentes méthodes d'entraînement du réseau convolutionnel du système *CNN* pour le corpus *OpenSAD 15* (segments de trois secondes). Le domaine source est le canal téléphonique (*src*).

| Méthode d'apprentissage | $EER_{avg}$ sur le domaine cible (%) | | | | | |
|---|---|---|---|---|---|---|
| | *B* | *D* | *E* | *F* | *G* | *H* |
| supervisé sur *source* | 57 | 52 | 48 | 51 | 30 | 50 |
| supervisé sur *cible* | **18** | 15 | 19 | 15 | 14 | 22 |
| distance entre les moyennes | 53 | 44 | 38 | 35 | 12 | 41 |
| *CORAL* | 32 | 32 | 26 | 18 | 11 | 20 |
| *MMD* | 19 | **11** | **16** | **13** | **9** | **18** |

Les trois fonctions de coût étudiées permettent d'améliorer la performance sur le domaine cible. Ensuite, si la *MMD* et *CORAL* atteignaient les mêmes performances lorsqu'on pouvait choisir les hyper-paramètres, les hyper-paramètres semblent plus robustes pour la *MMD*. C'est cette fonction de régularisation qui fonctionne le mieux pour les six canaux cibles. L'adaptation non supervisée avec la *MMD* permet d'atteindre une performance du même ordre qu'un apprentissage supervisé sur le domaine cible : un taux d'égale erreur moyen compris entre 11% et 19%.

### 4.3.6 Adaptation simultanée vers plusieurs canaux

Dans les expériences précédentes, nous nous sommes limités à l'adaptation entre un canal source et un unique canal cible. Peut-on par cette méthode entraîner un système qui fonctionnerait sur plusieurs canaux cibles en même temps ?

Nous proposons d'entraîner un système en ajoutant une fonction de régularisation qui tienne compte de tous les domaines cibles visés. La fonction de coût devient la suivante.

$$\mathcal{L} = \mathcal{L}_{classification}(R(\mathcal{D}_\mathcal{S})) + \lambda \sum_{\mathcal{D}_\mathcal{T} \in D} \mathcal{L}_{regularisation}(R_e(\mathcal{D}_S^{|\mathcal{X}}), R_e(\mathcal{D}_T^{|\mathcal{X}}))$$

où $D$ est l'ensemble fini des domaines cibles considérés.

Nous conservons les hyper-paramètres $\lambda$ et $\sigma^2$ sélectionnés dans la sous-partie précédente et réalisons un entraînement du système *CNN* en utilisant six domaines cibles correspondant aux six canaux radios du corpus *OpenSAD 15*.

TABLE 4.11 – Performance en terme de moyenne des taux d'égale erreur de différentes méthodes d'entraînement du réseau convolutionnel du système *CNN* pour le corpus *OpenSAD 15* (segments de trois secondes). Le domaine source est le canal téléphonique (*src*).

| Méthode d'apprentissage | $EER_{avg}$ sur le domaine cible (%) | | | | | |
|---|---|---|---|---|---|---|
| | *B* | *D* | *E* | *F* | *G* | *H* |
| supervisé sur *source* | 57 | 52 | 48 | 51 | 30 | 50 |
| supervisé sur chaque *cible* séparément | 18 | 15 | 19 | 15 | 14 | 22 |
| supervisé sur 7 canaux simultanément | 21 | 12 | 19 | 15 | 15 | 20 |
| MMD par canal | 19 | 11 | 16 | 13 | **9** | 18 |
| MMD simultanée vers 6 canaux | **14** | **8** | **12** | **10** | **9** | **16** |

Les performances de reconnaissance de la langue du tableau 4.11 révèlent que l'approche se généralise sans difficulté avec les mêmes hyper-paramètres pour l'ensemble des six domaines cibles.

### 4.3.7 Application au réseau d'identification de la langue d'un système *x-vector*

Nous appliquons maintenant la méthode d'adaptation de domaine sélectionnée à un système *x-vector* sur le corpus *RATS* comptant cinq langues. Cette méthode est appliquée au réseau d'identification de la langue qui est ensuite utilisé pour extraire des *x-vector*. Elle consiste en la régularisation avec la *MMD* sur la couche de sortie.

TABLE 4.12 – Performance de l'adaptation non supervisée du réseau d'identification de la langue dans le système *x-vector* sur le corpus *RATS*. Le corpus contient cinq langues et est constitué de segments de trois secondes. Une adaptation est réalisée pour chacun des canaux cibles. Le domaine source est le canal téléphonique (*src*).

| Méthode d'entraînement | $EER_{avg}$ sur le domaine cible (%) | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | *A* | *B* | *C* | *D* | *E* | *F* | *G* | *H* |
| supervisé sur *source* | 50,2 | 42,3 | 34,4 | 39,6 | 48,5 | 45,1 | 17,4 | 43,6 |
| supervisé sur *cible* | 14,6 | 12,5 | 12,6 | **6,7** | 13,6 | 13,5 | 8,6 | 14,2 |
| MMD | **12,7** | **10,6** | **11,7** | 7,6 | **13,3** | **11,9** | **5,5** | **12,2** |

Les résultats rapportés dans le tableau 4.12 montrent que l'adaptation de domaine non supervisée a fonctionné. Pour sept des huit canaux cibles testés, la régularisation avec la *Maximum Mean Discrepancy* permet d'obtenir une meilleure performance que l'apprentissage supervisé sur le domaine cible. Il est remarquable que la régularisation imposée dans l'espace des prédictions se traduise par une augmentation de l'invariance dans l'espace des *x-vector*. Ce point sera analysé plus précisément dans le chapitre 5.

### 4.3.8 Comparaison à d'autres méthodes d'adaptation de domaine

Comme expliqué dans le chapitre 3, les approches d'adaptation de domaine peuvent être réparties en deux familles : les approches *model based* et les approches *feature based*. Les approches *model based* consistent à modifier les paramètres du modèle appris sur le domaine source afin d'améliorer la performance sur le domaine cible. Les approches *feature based* apprennent une transformation entre les représentations des données des domaines source et cible afin de les faire coincider et que le modèle appris sur les données du domaine source transformées fonctionne bien sur le domaine cible.

D'autre part, un système de reconnaissance de la langue à l'état de l'art est constitué de trois modules principaux dont les paramètres peuvent être appris : un extracteur de *bottleneck features*, un extracteur de *x-vector* et un classifieur final. Sur lequel de ces modules devons-nous procéder à une adaptation non supervisée ?

Notre approche de régularisation de la fonction de coût du réseau *x-vector* est une adaptation *model based* du module central. Dans la littérature, au moment où nous commencions cette thèse, la majorité des travaux se concentraient sur des adaptations du classifieur final, tant *model based* que *feature based*. Nous avons proposé d'appliquer une approche *model based* d'adaptation non supervisée au réseau *x-vector*. Concomitamment, plusieurs travaux ont réalisé de telles adaptations du réseau *x-vector* pour différents types de variabilités, avec un apprentissage antagoniste [Rohdin *et al.*, 2019,Bhattacharya *et al.*, 2019b,Bhattacharya *et al.*, 2019a,Xia *et al.*, 2019,Meng *et al.*, 2019] ou avec une fonction de coût inspirée du transport optimal [Lu *et al.*, 2021].

Dans cette section, nous comparons notre méthode d'adaptation *model based* du réseau *x-vector* à une adaptation *feature based* du réseau *x-vector*, une adaptation *feature based* du classifieur final et une adaptation *model based* du classifieur final. Les méthodes d'adaptation comparées sont indiquées sur la figure 4.3 et portent des numéros pour faciliter la lecture. Notre méthode d'adaptation *model based* reposant sur une régularisation du réseau d'identification de la langue à l'aide de la *MMD* porte le numéro 4.

Nous utilisons la même transformation pour l'adaptation *feature based* du réseau *x-vector* et du classifieur final. Il s'agit de l'approche non supervisée *CORAL* [Sun et Saenko, 2016], largement utilisée [Alam *et al.*, 2018]. La transformation est une translation et une multiplication matricielle dans le but de faire coincider les deux premiers moments des distributions des *features* sur les deux domaines. Pour le réseau *x-vector*, cette transformation est appliquée à son entrée : les *bottleneck features* (méthode n°1). Pour le classifieur final, elle est appliquée aux *x-vector* (méthode n°2).

Pléthore de méthodes d'adaptation *model based* du classifieur final ont été proposées. Elles dépendent souvent étroitement de la nature du modèle, rendant les comparaisons

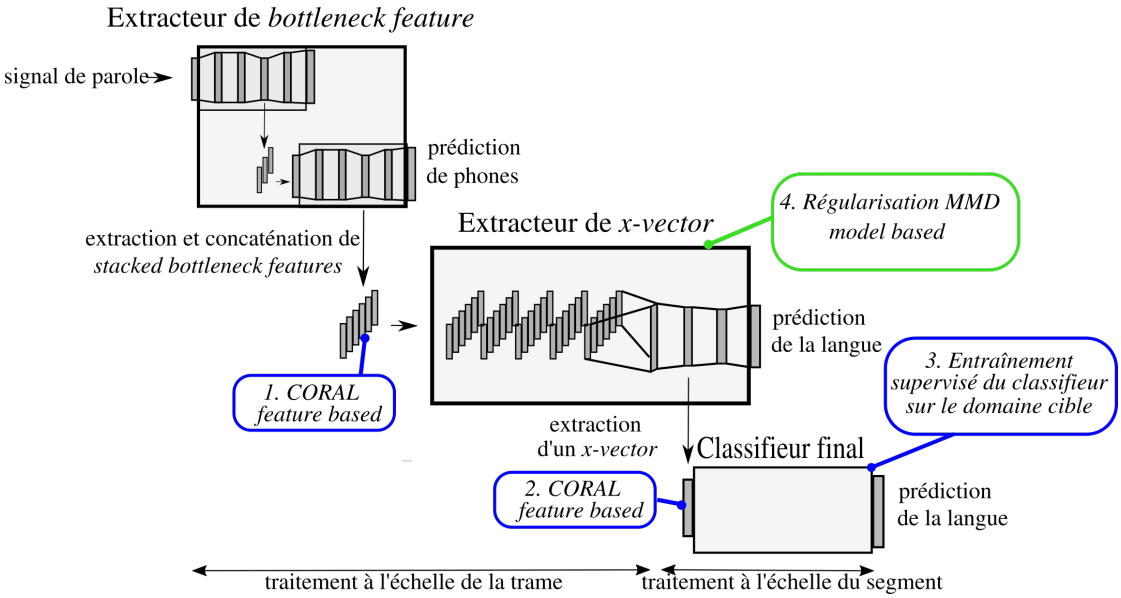



FIGURE 4.3 – Schéma du système de reconnaissance de la langue avec le positionnement des méthodes d'entraînement comparées. Chaque méthode porte un numéro pour faciliter la lecture du tableau 4.13.

difficiles. Afin d'évaluer la capacité maximale d'une adaptation *model based* du classifieur final, nous entraînons directement celui-ci de façon supervisée sur le domaine cible (méthode n°3).

Les performances de reconnaissance de la langue des différents systèmes sont rapportées dans le tableau 4.13. Les résultats pour tous les canaux cibles sont cohérents. Les méthodes d'adaptation auxquelles nous nous comparons permettent d'améliorer la performance par rapport à l'apprentissage supervisé sur le domaine source. Aucune n'atteint cependant la performance d'un système supervisé sur le domaine cible.

L'adaptation *feature based* permet un gain de performance assez faible, tant quand l'adaptation est appliquée au classifieur (méthode n°1) que quand elle est appliquée au réseau *x-vector* (méthode n°2). Cela n'est pas surprenant étant données les distorsions très importantes imposées au signal par les canaux radios. Une simple translation et une multiplication matricielle dans l'espace des *features* ne suffisent pas à compenser ces distorsions. Une meilleure performance pourrait peut-être être obtenue en utilisant une transformation plus sophistiquée. Il semble néanmoins difficile d'apprendre une transformation complexe de façon non supervisée. Remarquons toutefois que l'adaptation *feature based* fonctionne mieux quand elle est appliquée sur le réseau *x-vector* (méthode n°2) que sur le classifieur final (méthode n°1). Cela constitue un argument de plus qui conforte notre étude de l'adaptation du réseau *x-vector*.

Même entraîné sur le domaine cible (méthode n°3), le classifieur final ne parvient pas à atteindre la performance qu'il aurait obtenu avec des *x-vector* entraînés sur le domaine cible. C'est donc que des *x-vector* entraînés sur le domaine source ne permettent pas la

TABLE 4.13 – Performance de reconnaissance de la langue sur le corpus *RATS* en fonction des méthodes d'entraînement du réseau *x-vector* et du classifieur final. Le corpus contient cinq langues et est constitué de segments de trois secondes. Une adaptation est réalisée pour chacun des canaux cibles. Le domaine source est le canal téléphonique (*src*). Les numéros des méthodes font référence à la figure 4.3. On compte des méthodes supervisées (sup.) et des adaptations *feature based* (*feature b.*) et *model based* (*model b.*).

| Méthode d'entraînement | | | $EER_{avg}$ sur le domaine cible (%) | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| n° | réseau x-vector | classifieur final | *A* | *B* | *C* | *D* | *E* | *F* | *G* | *H* |
| | sup. sur *source* | sup. sur *source* | 50,2 | 42,3 | 34,4 | 39,6 | 48,5 | 45,1 | 17,4 | 43,6 |
| | sup. sur *cible* | sup. sur *cible* | 14,6 | 12,5 | 12,6 | 6,7 | 13,6 | 13,5 | 8,6 | 14,2 |
| 1 | sup. sur *source* | *CORAL feature b.* | 38,6 | 38,0 | 32,5 | 32,8 | 38,8 | 33,7 | 13,2 | 42,5 |
| 2 | *CORAL feature b.* | sup. sur *source* | 40,7 | 34,9 | 32,0 | 27,8 | 33,2 | 30,3 | 13,2 | 32,8 |
| 1, 2 | *CORAL feature b.* | *CORAL feature b.* | 39,5 | 35,3 | 31,9 | 28,0 | 32,5 | 29,8 | 13,2 | 32,7 |
| 3 | sup. sur *source* | sup. sur *cible* | 15,8 | 15,0 | 14,1 | 14,3 | 21,8 | 20,5 | 9,8 | 18,8 |
| 4 | *MMD model b.* | sup. sur *source* | 12,7 | 10,6 | 11,7 | 7,6 | 13,3 | 11,9 | 5,5 | 12,2 |
| 3, 4 | *MMD model b.* | sup. sur *cible* | **10,2** | **9,2** | **11,3** | **6,0** | **11,8** | **10,3** | **5,1** | **10,0** |

même capacité de discrimination sur le domaine cible que des *x-vector* entraînés sur le domaine cible. Cela motive notre étude de l'adaptation *model based* du réseau *x-vector*.

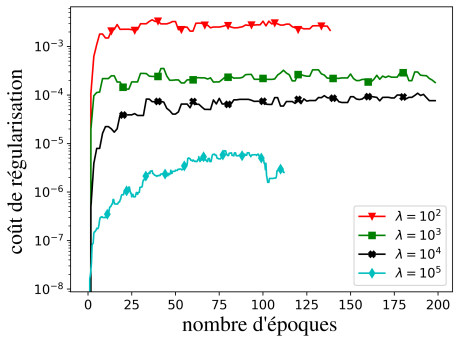
Parmi les méthodes comparées, l'adaptation *model based* du réseau *x-vector* avec la *MMD* est la plus performante (méthode n°4). Elle produit des *x-vector* dotés d'une grande invariance puisqu'un classifieur final entraîné sur le domaine source avec ces *x-vector* obtient de bonnes performances sur le domaine cible (un taux d'égal erreur moyen compris entre 5,5% et 13,3%), supérieures à un apprentissage supervisé sur le domaine cible pour sept des huit canaux considérés. L'objectif d'invariance permet donc non seulement de transférer la performance entre deux domaines mais également d'améliorer la performance de classification en produisant des représentations de meilleure qualité.

D'autre part, notons que les représentations produites par notre approche ne sont pas totalement invariantes. En effet, l'entraînement du classifieur final sur le domaine cible avec ces *x-vector* (méthodes n°3 et 4) améliore encore la performance, avec un taux d'égal erreur moyen compris entre 5,1% et 11,8%.

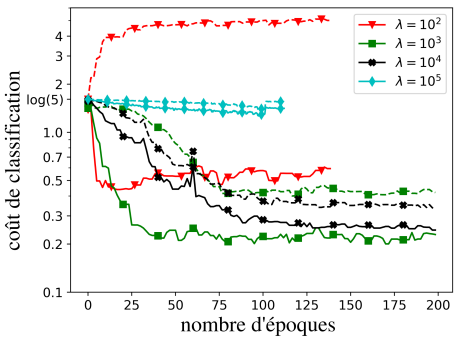
### 4.3.9 Effet des hyper-paramètres

La régularisation avec la *maximum mean discrepancy* dépend de deux hyper-paramètres : le poids $\lambda$ de la fonction de régularisation et la variance $\sigma^2$ du noyau Gaussien.

La variance $\sigma^2$ définit l'échelle des différences entre les domaines prise en compte par la *MMD*. Le réseau est initialisé avec des poids aléatoires. Aussi, au début de l'entraînement les distributions des activations sur les deux domaines sont semblables et la *MMD* quasiment nulle. Pendant l'entraînement, les domaines se séparent progressivement au fur et à mesure que les représentations deviennent plus discriminantes sur le domaine source. Quand l'échelle des différences entre les domaines atteint $\sigma^2$, cela active la *MMD*.




(a) Fonction de régularisation (*MMD* avec noyau Gaussien).




(b) Fonction de classification (entropie croisée). La ligne pleine correspond au domaine source, c'est la fonction de coût effectivement minimisée. La ligne en pointillés correspond au domaine cible.

FIGURE 4.4 – Valeurs des fonctions de coût au cours de l'entraînement du réseau d'identification de la langue du système *x-vector* avec la fonction de régularisation *MMD*, pour $\sigma^2 = 10$ et différentes valeurs de $\lambda$. Le canal téléphonique (*src*) est le domaine source, le canal *D* est le domaine cible. Les fonctions de coût sont mesurées sur l'ensemble de validation.

Par conséquent, si $\sigma^2$ est très petite, l'entraînement est contraint, les domaines ne se séparent pas mais la performance de classification ne s'améliore pas. À l'inverse, si $\sigma^2$ est trop grande, la *MMD* n'est jamais activée et il n'y a pas d'adaptation. En pratique, pour obtenir une convergence rapide, $\lambda$ doit être sélectionné en fonction de la valeur de $\sigma^2$. Nous avons observé que l'adaptation de domaine peut fonctionner avec des valeurs de $\sigma^2$ comprises entre 1 et $10^4$, sans différence manifeste dans cet intervalle de valeurs. Dans nos expériences, nous prenons $\sigma^2 = 10$.

Les figures 4.4a et 4.4b présentent la dynamique de l'entraînement pour différentes valeurs de l'hyper-paramètre $\lambda$ pour une adaptation du canal *src* vers le canal *D*. Les entraînements avec différentes valeurs de $\lambda$ ne durent pas le même nombre d'époques car la décroissance du pas d'apprentissage dépend de l'évolution de la fonction de coût sur l'ensemble de validation. Toutes les fonctions de coût sont mesurées sur l'ensemble de validation.

La figure 4.4a représente l'évolution de la *MMD* au cours de l'apprentissage. Pour tous les hyper-paramètres $\lambda$, elle augmente pour atteindre un plateau dont la valeur est contrôlée par $\lambda$. Le processus d'apprentissage atteint une configuration où la *MMD* est activée, ce qui signifie que la différence entre les domaines reste à une échelle d'ordre $\sigma^2$.

La figure 4.4b montre la fonction de classification sur les domaines source et cible. Des valeurs élevées de $\lambda$ ralentissent l'apprentissage sur le domaine source mais réduisent l'écart de performance entre les domaines. Avec de faibles valeurs de $\lambda$, le coût de classification sur le domaine cible diverge. Il est notable que pour $\lambda = 100$, l'adaptation non

supervisée ne fonctionne pas mais empêche aussi la convergence sur le domaine source. Pour des valeurs plus élevées, le choix de $\lambda$ est un compromis entre l'écart de performance entre les domaines et la durée du processus d'entraînement sur le domaine source. En pratique, pour de trop grandes valeurs de $\lambda$, l'apprentissage est si contraint que la convergence sur le domaine source ne peut pas advenir.

Les hyper-paramètres sélectionnés lors d'une adaptation entre les canaux $G$ et $D$ sont $\lambda = 10^4$ et $\sigma^2 = 10$. Ces hyper-paramètres ont été appliqués à toutes les configurations d'adaptation pour tous les canaux cibles. En effet, un scénario d'adaptation non supervisée n'autorise pas la sélection des meilleurs hyper-paramètres pour chaque configuration puisque la performance sur le domaine cible ne peut être estimée, faute de données annotées. Par conséquent la robustesse des hyper-paramètres aux différentes configurations est indispensable pour une méthode d'adaptation non supervisée. Nous avons vérifié que les hyper-paramètres sélectionnés donnaient d'excellentes performances d'adaptation non supervisée pour huit configurations avec différents canaux radios.

### 4.3.10 Conclusion sur l'adaptation non supervisée

L'adaptation de domaine non supervisée est un scénario de grand intérêt puisqu'il ne nécessite pas d'annotation de données du domaine cible. Nous avons montré que la régularisation de la fonction de coût du réseau d'identification de la langue avec une fonction mesurant l'invariance des distributions des représentations entre les domaines permettait de résoudre cette tâche d'adaptation non supervisée. Nous l'avons vérifié expérimentalement pour deux systèmes et deux corpus correspondant à neuf canaux de transmission. Des résultats complémentaires pour un troisième système sont présentés en annexe B.

L'approche sélectionnée est l'utilisation de la *maximum mean discrepancy* sur la couche de sortie du réseau. Elle permet d'augmenter l'invariance au canal des scores, mais aussi des représentations extraites d'une couche intermédiaire. Les hyper-paramètres de cette approche sont robustes aux configurations d'adaptation étudiées. Les représentations produites ne sont cependant pas totalement invariantes. Deux pistes de prolongement possible de ce travail se dégagent : combiner l'adaptation non supervisée du réseau *x-vector* avec une adaptation non supervisée du classifieur, ou augmenter encore l'invariance des *x-vector* en raffinant notre méthode.

D'autre part, pour un système *x-vector*, l'approche proposée est supérieure aux adaptations *feature based* du réseau *x-vector* et du classifieur final, et à l'adaptation *model based* du classifieur final. Il serait intéressant de réaliser la même étude pour le réseau chargé de l'extraction des *bottleneck features*, dans le même esprit que des travaux existants qui explorent d'autres sources de variabilité, comme la variabilité due au locuteur [Peng *et al.*, 2019].

Ainsi, pour une adaptation non supervisée, une classification efficace sur le domaine cible peut être atteinte par un alignement des deux domaines dans un espace de représentations. Ces représentations invariantes sont de bonne qualité et l'invariance permet également d'améliorer la classification sur le domaine source. Cependant, une limite essentielle de ce cadre expérimental est la présence des mêmes langues dans les mêmes

proportions sur les canaux source et cible (soit l'absence de *label shift*), ce qui rend théoriquement possible un alignement parfait. Une configuration où les distributions des langues sont différentes sur les deux canaux est l'objet du scénario étudié dans la section suivante.

## 4.4 Apprentissage multi-domaines

L'apprentissage multi-domaines est un scénario d'apprentissage supervisé où plusieurs domaines sont présents dans l'ensemble d'entraînement. Au sein d'un système *x-vector*, nous montrons que la régularisation du réseau d'identification de la langue peut augmenter l'invariance des représentations entre les domaines et donc améliorer la performance de classification. Deux méthodes de régularisation sont explorées : la réduction de l'écart entre les distributions des représentations des différents domaines et le *metric learning*.

### 4.4.1 Corpus utilisé

Nous utilisons pour l'évaluation expérimentale de ce scénario les conditions de l'évaluation *NIST LRE 2011*. Les corpus d'entraînement et de test comportent deux canaux de transmission : le téléphone et des appels téléphoniques retransmis lors d'émissions de radio ou de télévision. Ce dernier canal est appelé *BNBS* (*broadcast narrowband speech*). Téléphone et *BNBS* seront nos deux domaines. De plus, le corpus d'entraînement compte cinquante et une langues, dont vingt-quatre sont utilisées pour l'évaluation de la performance du système. Les corpus sont décrits en annexe A.1.2. Le corpus d'entraînement présente un *label shift* où les distributions des langues diffèrent fortement pour les deux canaux. La distribution des langues par canal pour les ensembles d'entraînement et de test est décrite dans le tableau A.4. Ainsi, seules treize langues sont présentes sur les deux canaux dans l'ensemble d'entraînement, tandis que dix-neuf sont observées uniquement sur le canal téléphonique et dix-neuf uniquement pour le canal *BNBS*. À l'inverse, parmi les vingt-quatre langues utilisées pour l'évaluation du système, vingt sont évaluées sur les deux canaux alors que seules sept d'entre elles ont été observées sur les deux canaux dans l'ensemble d'entraînement.

Le cadre de l'évaluation *NIST LRE 2011* imposait de produire un rapport de vraisemblance pour toutes les paires de langues cibles et non cibles, soit $\frac{24\times23}{2} = 276$ paires de langue. La métrique de référence est *APD*, la moyenne des coûts de détection pour un ensemble de paires de langues. Ces paires de langues sont les vingt-quatre paires pour lesquelles le $minDCF$ du système est le plus élevé pour l'ensemble de test de durée nominale trente secondes. Autrement dit, le système est évalué pour les vingt-quatre paires de langues pour lesquelles il commet le plus d'erreurs. Le point de fonctionnement choisi correspond aux paramètres $C_{FA} = C_{FR} = 1$ et $P_{cible} = 0{,}5$. Nous calculons également les métriques $C_{avg}$ pour le même point de fonctionnement, $EER$, ainsi que les coûts de détection $minAPD$ et $minDCF$ faisant abstraction des erreurs de calibration.

Dans ce dispositif expérimental, nous ne disposons que de deux canaux de transmission. Nous explorons donc l'apprentissage multi-domaines uniquement dans une configu-

ration avec deux domaines.

### 4.4.2 Système utilisé

Nous réalisons les expériences pour ce scénario avec le système *x-vector bis* décrit dans le tableau 4.14. Trois éléments ont changé par rapport au système *x-vector* utilisé dans le scénario d'adaptation de domaine non supervisée (tableau 4.5). D'abord nous utilisons des augmentations de données classiques pour l'entraînement du réseau d'identification de la langue, dans le but de reproduire une recette standard d'entraînement de système *x-vector* [Snyder *et al.*, 2018]. Nous modifions également le classifieur final en choisissant un classifieur Gaussien plutôt qu'un *SVM*. Cela est dû au mode d'évaluation pour le corpus *NIST LRE 2011* qui exige de produire un rapport de vraisemblance pour chaque paire de langues cible et non cible. Cela est plus aisé avec un modèle génératif comme un classifieur Gaussien qu'avec un modèle discriminant. Enfin, nous introduisons un module de calibration puisque nous évaluons le système avec les métriques $APD$ et $C_{avg}$.

### 4.4.3 Réduction de l'écart entre les domaines

Notons $\mathcal{D}_T$ et $\mathcal{D}_B$ les deux domaines téléphoniques et *BNBS*. En premier lieu, nous évaluons la méthode de régularisation développée pour l'adaptation de domaine non supervisée pour ce scénario d'apprentissage multi-domaines. Nous entraînons le réseau d'identification de la langue avec la fonction de coût suivante.

$$\mathcal{L} = \mathcal{L}_{classification}(R(\mathcal{D}_T)) + \mathcal{L}_{classification}(R(\mathcal{D}_B)) + \lambda \mathcal{L}_{\text{régularisation}}(R_e(\mathcal{D}_T^{|\mathcal{X}}), R_e(\mathcal{D}_B^{|\mathcal{X}}))$$

Afin de limiter la recherche d'hyper-paramètres, nous utilisons pour $\mathcal{L}_{\text{régularisation}}$ la *maximum mean discrepancy* avec le noyau associé $k_{\text{énergie}}$. De plus, dans ce cadre expérimental, une recherche d'hyper-paramètre pour le poids $\lambda$ de la régularisation fausserait les résultats puisque nous ne disposons que d'une seule condition expérimentale pour l'évaluation. Nous nous limitons donc à la valeur $\lambda = 1$. Enfin, nous choisissons de ne pas appliquer la régularisation sur la couche de sortie mais dans l'espace des *x-vector*. Deux raisons président à ce choix. D'abord la contrainte d'invariance dans l'espace des prédictions est irréaliste dans le cas d'un corpus non parallèle où nous souhaitons que les distributions des scores correspondent aux distributions des langues. Nous supposons que la contrainte d'invariance dans l'espace des *x-vector* laisse la liberté au modèle d'utiliser les couches hautes du réseau pour produire une classification optimale pour chacun des deux domaines. D'autre part, nous appliquons la contrainte dans l'espace des *x-vector* afin de permettre une comparaison équitable avec les fonctions de coût de *metric learning* utilisées dans la suite de cette partie et qui opèrent naturellement dans l'espace des *x-vector*.

Les performances du système entraîné avec l'entropie croisée et du système régularisé avec la *MMD* sont présentées dans le tableau 4.15. L'approche par régularisation a permis d'améliorer les performances pour les segments de dix et trente secondes mais pas pour les segments de trois secondes. Le gain en performance est clair en terme de $C_{avg} \times 100$,

TABLE 4.14 – Architecture du système *x-vector bis*.

| Extraction de *bottleneck features* multilingues, calculés à partir de spectrogrammes en échelle *Mel*, puis production d'un *x-vector* par le *TDNN* entraîné pour l'identification de la langue, fourni en entrée d'un classifieur Gaussien qui prédit la langue | | |
|---|---|---|
| Module | Modèle | Hyper-paramètres |
| détection d'activité vocale | réseau convolutionnel, système décrit en annexe C | il fonctionne par trame et est entraîné sur le corpus *RATS SAD* |
| *features* acoustiques | log-spectrogrammes en échelle *Mel* | $\Delta_F = 25ms$, $\delta_F = 10ms$, prétraitement décrit dans [Silnova *et al.*, 2018] |
| augmentations de données | quatre augmentations | réverbération (avec le corpus *RIR*), bruit additif, musique (corpus *MUSAN*) et *babble noise* |
| *bottleneck features* | *stacked bottleneck features* | modèle multilingue produisant des *features* de dimension 80. Nous utilisons le modèle de *BUT / Phonexia* [Silnova *et al.*, 2018] entraîné à prédire des triphones pour 17 langues du corpus *Babel*. |
| modélisation du segment | *TDNN* standard [Snyder *et al.*, 2018] | entraîné pour l'identification de la langue avec différentes fonctions de coût avec des segments de durée comprise entre deux et quatre secondes, avec les quatre augmentations de données |
| classifieur final | classifieur Gaussien | *LDA*, centrage, normalisation $L_2$ des *embeddings* et prédiction de scores par un classifieur Gaussien avec une matrice de covariance partagée entre les classes |
| calibration | régression logistique | utilisation de l'outil *Focal* [Brümmer, 2007] |

TABLE 4.15 – Performance du système *x-vector bis* sur les ensembles de test du corpus *NIST LRE 2011* en fonction de la fonction de coût d'entraînement du réseau de neurones d'identification de la langue. Cinq fonctions de coût sont utilisées : l'entropie croisée (*CE*), *additive angular margin softmax* (*AAM*), la *triplet loss*, la *n-pair loss* et la *maximum mean discrepancy* (*MMD*).

| | Performance (%) pour les différentes durées de test | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 30 secondes | | | | | 10 secondes | | | | | 3 secondes | | | | |
| Fonction de coût | APD | | DCF | | EER | APD | | DCF | | EER | APD | | DCF | | EER |
| | min | | min $C_{avg}$ | | | min | | min $C_{avg}$ | | | min | | min $C_{avg}$ | | |
| *triplet loss* | 12,5 | 15,9 | 5,7 | 9,3 | 6,1 | 19,9 | 23,5 | 9,6 | 12,2 | 10,2 | 29,7 | 32,4 | 19,1 | 20,5 | 19,6 |
| *n-pair loss* | 9,8 | 12,8 | 3,8 | 6,9 | 4,2 | 15,6 | 20,6 | 6,6 | 9,1 | 7,1 | 22,6 | 27,9 | 13,6 | 15,0 | 14,0 |
| *CE* | 6,5 | **8,7** | 2,8 | 5,7 | 3,1 | 11,7 | 15,3 | 5,2 | 7,1 | 5,5 | **20,2** | **22,9** | **11,3** | **12,4** | **11,6** |
| *CE* et *MMD* | 6,4 | **8,7** | 2,7 | **4,2** | **3,0** | 11,1 | **14,2** | 4,8 | **6,2** | 5,2 | 21,2 | 23,9 | 12,6 | 13,7 | 13,0 |
| *CE* et *n-pair* | **6,2** | 9,2 | **2,6** | 5,3 | **3,0** | **10,8** | 15,3 | **4,7** | **6,2** | **5,1** | 21,0 | 24,7 | 12,2 | 13,8 | 12,7 |
| *AAM* | 4,8 | **6,6** | **2,1** | 4,8 | **2,4** | 9,3 | 11,5 | 4,2 | 6,1 | 4,6 | 19,1 | 21,6 | 11,7 | **13,0** | 12,1 |
| *AAM* et *n-pair* | **4,5** | 6,9 | **2,1** | **4,6** | **2,4** | **8,8** | **11,4** | **4,1** | **5,8** | **4,5** | **18,2** | **21,1** | **11,6** | **13,0** | **12,0** |

avec un passage de 7,1 à 6,2 par l'ajout de la régularisation pour des segments de dix secondes. La stratégie de construction de représentations invariantes a donc fonctionné. Nous verrons dans le chapitre 5 que les *x-vector* produits ont bien gagné en invariance.

### 4.4.4 Regrouper les *x-vector* par langue

Alors que la *MMD* vise à produire des représentations (ici *x-vector*) ayant des distributions similaires entre canaux, le *metric learning* s'emploie à regrouper les représentations d'une même classe, et ce indépendamment du canal. Ici nous entraînons le réseau d'identification de la langue avec deux fonctions de coût de *metric learning* : *triplet loss* et *n-pair loss*. Comme la classification par le classifieur final opère en fonction de proximités géométriques dans l'espace des *x-vector*, c'est dans cet espace que nous appliquons les fonctions de *metric learning*.

Les performances associées à ces fonctions de coût sont rapportées dans le tableau 4.15. Elles ne permettent pas d'égaler l'entropie croisée dans le cadre du système étudié. La *n-pair loss* est plus efficace que la *triplet loss*, avec par exemple un taux d'égale erreur de 14,0% au lieu de 19,6% pour des segments de trois secondes. Une piste d'amélioration abondamment traitée dans la littérature serait le développement d'une stratégie d'échantillonnage des exemples négatifs.

### 4.4.5 Régularisation par *metric learning*

Convaincus de l'intérêt du *metric learning* pour regrouper les échantillons par langue dans l'espace des *x-vector* mais confrontés au fait que les fonctions de coût de *metric learning* utilisées ne permettent pas de rivaliser avec une fonction de classification, nous décidons d'utiliser le *metric learning* comme une fonction de régularisation. Nous sui-

vons donc le même modèle que pour l'entraînement avec la *MMD*. La fonction de coût minimisée est la suivante.

$$\mathcal{L} = \mathcal{L}_{classification}(R(\mathcal{D}_T)) + \mathcal{L}_{classification}(R(\mathcal{D}_B)) + \lambda \mathcal{L}_{metric}(R_e(\mathcal{D}_T \cup \mathcal{D}_B))$$

où $\mathcal{D}_T \cup \mathcal{D}_B$ correspond à la réunion des deux domaines.

Pour les mêmes raisons que pour les expériences avec la *MMD*, nous nous limitons à la valeur $\lambda = 1$. Nous évaluons l'intérêt de la régularisation par le *metric learning* pour deux fonctions de classification : l'entropie croisée (*CE*) et *additive angular margin softmax* (*AAM*). Nous utilisons la *n-pair loss* en raison de sa meilleure performance que la *triplet loss*.

Les performances sur le corpus *NIST LRE 2011* rapportées dans le tableau 4.15 montrent que la régularisation du réseau d'identification de la langue avec la *n-pair loss* améliore légèrement la classification, tant pour l'entropie croisée (un $C_{avg} \times 100$ de 6,2 au lieu de 7,1 pour des segments de dix secondes) que pour *AAM* (un $C_{avg} \times 100$ de 5,8 au lieu de 7,1 pour des segments de dix secondes).

Le gain de performance pour l'entropie croisée est du même ordre que celui obtenu avec la régularisation entre canaux obtenue avec la *MMD* (avec des $C_{avg} \times 100$ de 6.2 pour des segments de dix secondes). La *n-pair loss* en revanche n'utilise pas l'information de canal. C'est donc une méthode plus puissante dans le cadre de l'apprentissage multi-domaines. Nous supposons (et vérifierons dans le chapitre 5) qu'elle a permis de limiter l'impact de variabilités présentes dans l'ensemble d'entraînement et ce de manière implicite, contrairement à la *MMD*.

### 4.4.6 Conclusion sur l'apprentissage multi-domaines

À l'issue de ces expériences, nous concluons que la construction de représentations invariantes au moyen de la régularisation de la fonction de coût d'identification de la langue a un intérêt pour un scénario d'apprentissage multi-domaines. Le fait que les distributions des langues soient différentes pour différents domaines n'empêche pas la constitution d'une certaine invariance par ces méthodes.

Deux types de fonctions de coût permettent d'augmenter l'invariance avec des performances comparables : la *maximum mean discrepancy* comme fonction d'alignement de deux distributions et la *n-pair loss* comme fonction de *metric learning*. La *MMD* permet d'augmenter l'invariance par rapport à un type de variabilité, *a priori* connue. Dans notre travail, il s'agit du canal de transmission. La *n-pair loss* ne se concentre pas sur une variabilité précise, elle ne nécessite pas d'étiquettes de domaines et regroupe les représentations par langue, réduisant *de facto* l'impact des variabilités présentes dans l'ensemble d'entraînement. En revanche la *n-pair loss* est une fonction de coût supervisée utilisant des étiquettes de langues. Elle ne peut donc pas être appliquée dans un scénario d'adaptation non supervisée.

## 4.5 Généralisation à un domaine inconnu

### 4.5.1 Description du scénario

Le scénario de généralisation à un nouveau domaine est le plus difficile. Nous supposons disposer d'un corpus d'entraînement contenant plusieurs domaines comme dans la partie précédente. Contrairement à l'apprentissage multi-domaines, l'objectif n'est plus la maximisation de la performance sur les domaines observés. Au contraire le but du scénario est la bonne performance sur un nouveau domaine non observé lors de l'apprentissage.

Ce problème est impossible à aborder sans faire d'hypothèse sur le lien entre le nouveau domaine et les domaines observés au cours de l'apprentissage [Arjovsky, 2020]. Dans le cas du changement de canal de transmission, remarquons que les distorsions imposées au signal, bien que complexes, préservent le contenu linguistique. Les transformations possibles sont nombreuses mais très limitées et structurées.

Dès lors, supposons disposer de plusieurs domaines dans l'ensemble d'entraînement appartenant à cet espace de transformations admissibles. Nous avons vu dans les parties précédentes qu'il était possible d'augmenter l'invariance des représentations à ces transformations observées. Une invariance aux domaines observés lors de l'apprentissage peut-elle induire une invariance à un nouveau domaine non observé lors de l'apprentissage ? Des résultats théoriques ont été démontrés pour ce problème qui donnent des garanties sur l'invariance des représentations à un domaine inconnu en faisant des hypothèses sur la complexité de l'ensemble des domaines admissibles et de la couverture de cet ensemble par les données d'entraînement [Redko *et al.*, 2020, Arjovsky, 2020]. Dans notre travail, nous nous limitons à une exploration expérimentale de cette question pour la généralisation à un canal inconnu d'un système de reconnaissance de la langue.

Le scénario de généralisation à un nouveau domaine présente une difficulté de taille. Comment échantillonner un corpus de test d'un canal de transmission véritablement inconnu ? La tâche 1 de l'évaluation *Oriental Language Recognition 2020* a été l'occasion d'un test en conditions réelles où nous ne disposions d'aucune information sur les données de test au cours de la conception du système.

### 4.5.2 Corpus utilisé

Pour cette série d'expériences, nous nous plaçons dans le cadre de la tâche 1 du *challenge OLR 2020*. Nous ne disposons que de deux informations sur le corpus de test. Il contient six langues connues : cantonais, coréen, indonésien, japonais, russe et vietnamien ; et il correspond à des canaux de transmission inconnus.

Le corpus d'entraînement est imposé par les règles de la compétition. Il comprend seize langues. La majorité du corpus provient de téléphones mobiles. Seules six langues sont présentes sur d'autres canaux, également inconnus et *a priori* différents des canaux de test. Parmi ces six langues, seules trois font partie des langues cibles appartenant au corpus de test.

Nous utilisons les corpus introduits dans la sous-partie 4.1.1 et décrits en annexe A.3.

Tous les systèmes sont entraînés avec le corpus multi-domaines *entraînement tous canaux* correspondant à la phase appelée *conception du système final* en annexe A.3. Nous mesurons la performance pour quatre ensembles de test : l'ensemble de validation correspondant à la même distribution que l'ensemble d'entraînement et sur lequel s'effectuent calibration et fusion des systèmes, les corpus *test mobile* et *test mobile et canaux inconnus* décrits dans la sous-partie 4.1.1 et le corpus d'évaluation de la tâche 1 du *challenge OLR 2020* correspondant à des canaux inconnus.

### 4.5.3 Systèmes utilisés

Nous utilisons les deux systèmes décrits dans la sous-partie 4.1.1, l'un basé sur un *GMM* (tableau 4.1), l'autre sur un réseau de neurones d'identification de la langue (tableau 4.2). Nous utilisons deux fonctions de classification pour l'entraînement du réseau d'identification de la langue : l'entropie croisée (CE) et *additive angular margin softmax* (AAM). De plus, lorsque plusieurs systèmes sont fusionnés, la fusion est réalisée lors de l'étape de calibration au niveau des scores et sur l'ensemble de validation de même distribution que l'ensemble d'entraînement.

### 4.5.4 Stratégie adoptée

Nous évaluons la méthode de régularisation du réseau d'identification de la langue dans l'espace des *x-vector*. L'objectif de cette approche est l'obtention d'une invariance entre les canaux de l'ensemble d'entraînement qui se généralise aux canaux inconnus de l'ensemble de test. Pour la régularisation, nous utilisons les mêmes fonctions de coût que pour l'apprentissage multi-domaines : la *MMD* entre le canal téléphonique et les canaux inconnus et la *n-pair loss*. Une fois de plus, nous utilisons le noyau $k_{\text{énergie}}$ pour la *MMD*.

### 4.5.5 Analyse des performances

Les performances de chacun des systèmes sont rapportées dans le tableau 4.16 en terme de $C_{avg}$, la métrique d'évaluation des systèmes lors du *challenge OLR 2020*.

Notons d'abord que la meilleure performance individuelle est atteinte par des systèmes différents en fonction des corpus de test. Sur le corpus de validation, le système standard entraîné avec l'entropie croisée atteint la meilleure performance ($C_{avg} \times 100 = 2{,}50$). Il s'agit de la meilleure recette pour un corpus de test similaire au corpus d'apprentissage. Sur le canal mobile connu, le *GMM* atteint la meilleure performance (2,53). C'est donc que cette approche classique reste très efficace pour des données présentant peu de variabilité. Enfin, le système entraîné avec *AAM* atteint la meilleure performance sur le corpus de test correspondant à des canaux inconnus (5,34).

Les réseaux régularisés n'atteignent la meilleure performance sur aucun corpus. En revanche, ils ont acquis une forme de robustesse puisque l'écart entre les performances sur les corpus de test *test mobile* et *test mobile et canaux inconnus* est considérablement réduit. Ces systèmes robustes ont un véritable intérêt pour la généralisation à un nouveau domaine. En effet, pour les conditions connues, une bonne performance est atteinte par

TABLE 4.16 – Performance de reconnaissance de la langue pour la tâche 1 de la compétition *OLR 2020* en termes de $C_{avg}$. La performance est mesurée pour les trois ensembles de développement, *validation* (val.), *test mobile* (mob.) et *test mobile et canaux inconnus* (can.) ainsi que l'ensemble d'évaluation de la compétition (eval).

| Systèmes | | Tâche 1 $C_{avg} \times 100$ | | | |
|---|---|---|---|---|---|
| | | val. | mob. | can. | eval |
| systèmes individuels | entropie croisée (CE) | **2,50** | 4,69 | 7,26 | 5,12 |
| | **AAM** | 3,68 | 3,14 | **5,34** | **5,08** |
| | CE et n-pair $\lambda = 1$ | 3,22 | 5,73 | 7,25 | 7,52 |
| | CE et MMD $\lambda = 1$ | 3,18 | 5,39 | 7,87 | 5,54 |
| | CE et MMD $\lambda = 100$ | 4,01 | 5,89 | 7,60 | 7,61 |
| | GMM | 3,31 | **2,53** | 10,82 | 7,13 |
| fusion de systèmes | AAM - GMM | 1,37 | 1,54 | 4,37 | 2,82 |
| | AAM - GMM - MMD 100 | 0,98 | 1,57 | 3,91 | 2,47 |
| | **AAM - GMM - MMD 100 - n-pair - MMD 1** | **0,93** | **1,53** | **3,67** | **2,28** |

la fusion du *GMM* avec le système *AAM* (1,54). Pour les canaux inconnus, un gain significatif est apporté par la fusion avec les trois réseaux régularisés (3,67).

Après que les étiquettes de l'ensemble d'évaluation aient été publiées par les organisateurs de la compétition *OLR 2020*, nous avons pu évaluer les systèmes sur l'ensemble d'évaluation. Les résultats sur ce corpus (tableau 4.16) sont cohérents avec la performance obtenue sur le corpus *test mobile et canaux inconnus*. La combinaison des trois réseaux régularisés apporte un gain de performance aux systèmes *AAM* et *GMM*. C'est la fusion de ces cinq systèmes qui a atteint la meilleure performance pour les tâches 1 et 3 du *challenge OLR 2020*.

### 4.5.6 Conclusion sur la généralisation à un nouveau domaine

Le scénario de généralisation à un canal inconnu pose la question d'un cadre expérimental fiable où l'information sur le canal de test est véritablement cachée au concepteur du système. C'était le cas lors de l'évaluation *Oriental Language Recognition 2020*. Pour ce scénario, nous constatons que la régularisation du réseau d'identification de la langue ne permet pas à elle seule d'améliorer la généralisation à un canal inconnu pour un système individuel.

En revanche, cette approche permet d'augmenter la robustesse au canal de modèles individuels, parfois aux dépens de la performance sur des conditions connues. Cela trouve son intérêt dans le cadre de la fusion de systèmes où un système robuste peut apporter un vrai gain à un système plus performant sur des conditions connues mais ne disposant pas de représentations invariantes.

Plus largement, la régularisation du réseau d'identification de la langue trouve sa place parmi d'autres outils ayant fait preuve de leur intérêt pour la généralisation à un

domaine inconnu : l'utilisation de *features* robustes, des augmentations de données, une méthode de sélection de paramètres robustes au cours de la descente de gradient et la fusion de systèmes. Le problème de la généralisation à un canal inconnu reste largement ouvert.

## 4.6 Conclusion

Partant du constat que les systèmes de reconnaissance de la langue ne sont pas robustes à un changement de canal de transmission, nous montrons que cette caractéristique est principalement explicable par l'absence ou la sous-représentation des canaux d'intérêt dans le corpus d'entraînement. Nous tentons donc de produire des systèmes utilisant des représentations invariantes au canal du signal d'entrée. Nous nous concentrons sur les réseaux de neurones d'identification de la langue, module central de nombreux systèmes de reconnaissance de la langue. Nous proposons d'imposer la contrainte d'invariance au canal à ces réseaux en appliquant une régularisation de leur fonction de coût. Deux types de fonctions de régularisation peuvent améliorer l'invariance : les fonctions visant à un alignement entre les domaines et les fonctions de coût de type *metric learning*. Les premières sont très efficaces pour réduire l'impact d'une variabilité connue avec des données non annotées en langue, tandis que les secondes peuvent réduire l'impact de variabilités *a priori* inconnues, mais nécessitent des étiquettes de langues.

Nous montrons dans trois scénarios expérimentaux différents l'intérêt de la construction de représentations invariantes : adaptation de domaine non supervisée, apprentissage multi-domaines et généralisation à un nouveau domaine. L'amélioration de la performance pour ces scénarios mettant en jeu un changement de canal de transmission justifie l'hypothèse qui sous-tend notre méthode : la perte de performance est due à une variabilité trop grande des représentations au canal de transmission. Dans le chapitre 5, nous analysons plus finement ce phénomène en mesurant directement les variabilités dans les espaces de représentation.

Les travaux menés dans ce chapitre peuvent être étendus. Trois directions principales émergent.

D'abord l'évaluation expérimentale des approches proposées peut être complétée par d'autres scénarios. L'adaptation de domaine supervisée pourrait être étudiée ainsi que des scénarios d'apprentissage multi-domaines avec plus de deux domaines. D'autre part il serait intéressant de faire varier le nombre de langues partagées entre les deux domaines dans l'ensemble d'entraînement, qui est un des critères principaux de la difficulté d'un alignement des représentations. Enfin, la généralisation à un nouveau domaine mériterait d'être analysée en faisant varier la proximité des canaux de test avec les canaux vus lors de l'entraînement, probablement dans une configuration plus maîtrisée que la participation à une compétition pour laquelle nous avons peu d'informations sur le canal d'évaluation.

D'autre part, d'autres fonctions de coût s'imposent dans la littérature avec le même objectif d'augmentation de l'invariance des représentations. Les approches les plus répandues sont des apprentissages antagonistes et des fonctions de coût inspirées du transport

optimal. Il serait pertinent de les comparer avec la *maximum mean discrepancy* dans le cas de la robustesse au canal de transmission de systèmes de reconnaissance de la langue.

Enfin, nous montrons dans ce travail le gain en performance obtenu en augmentant l'invariance des représentations produites par le réseau de neurones central du système. Nous vérifions dans le cas de l'adaptation de domaine non supervisée qu'une telle approche est supérieure à l'adaptation du classifieur final. Il serait donc pertinent d'augmenter l'invariance sur un module encore antérieur du système : l'extracteur de *bottleneck features*.

# 5

# Analyse des systèmes

Nous avons proposé une méthode d'augmentation de la robustesse au canal d'un système de reconnaissance de la langue. Celle-ci repose sur l'idée d'utiliser des réseaux de neurones profonds pour construire des représentations du signal invariantes au canal de transmission. Dans ce chapitre, nous nous intéressons à la mesure de l'impact d'un facteur de variabilité dans ces espaces de représentations. Nous visualisons d'abord les phénomènes que nous souhaitons mesurer avec des projections en deux dimensions de l'espace de représentations. Nous explicitons ensuite l'intérêt d'une mesure directe de la variabilité des représentations qui apporte une information complémentaire à la simple mesure de performance d'un système sur différents ensembles de test et qui est plus générale que l'entraînement et l'évaluation de plusieurs classifieurs dans un même espace de représentations. Une première manière de mesurer l'impact global d'un facteur de variabilité sur le système est le rapport entre les covariances inter-classes et intra-classes pour différents facteurs de variabilité. Des mesures de divergence entre groupes de vecteurs dans l'espace des représentations permettent une analyse plus fine. Nous utilisons ces deux outils pour comprendre l'effet des fonctions de régularisation étudiées dans le chapitre précédent sur un système de reconnaissance de la langue.

## 5.1 Configurations de l'espace des représentations

Nous avons obtenu une amélioration de la performance de systèmes de reconnaissance de la langue en imposant des contraintes de proximité dans l'espace des représentations. Dans ce chapitre, nous visons à mesurer ces propriétés. Dans le but de mieux les comprendre, nous visualisons d'abord l'espace des représentations par des projections dans un espace à deux dimensions.

### 5.1.1 Visualisations de l'espace des représentations

La visualisation des représentations s'obtient en projetant dans un espace à deux dimensions les *x-vector* représentatifs des segments de parole d'un corpus de test. Ces vecteurs ont généralement une dimension élevée (entre 128 et 512). Des couleurs ou sym-

boles peuvent être utilisés pour représenter les étiquettes associées aux échantillons. Trois types de visualisations sont couramment utilisés. L'analyse en composantes principales (*ACP*) [Shlens, 2014] est une projection linéaire non supervisée sur le plan portant la plus grande proportion de la variance de la distribution. L'analyse discriminante linéaire (*linear discriminant analysis - LDA*) [Tharwat *et al.*, 2017] est une projection linéaire supervisée. Elle sélectionne le plan le plus discriminant pour la classification, en fonction d'un ensemble d'étiquettes associées aux données. La *t-SNE* (*t-distributed Stochastic Neighbor Embedding*) [Van Der Maaten et Hinton, 2008] est une visualisation non linéaire tentant de respecter la distribution des distances entre paires d'échantillons. Elle rend compte des proximités locales dans l'espace des représentations.

L'*ACP* a l'avantage d'être non supervisée mais elle ne rend compte que des variations du premier ordre dans la distribution des représentations. Elle n'est en général pas assez fine pour observer les phénomènes qui nous intéressent : la variabilité due aux langues et aux canaux de transmission. La *LDA* est linéaire donc facilement interprétable. De plus son caractère supervisé permet de sélectionner l'information d'intérêt. La *t-SNE* est non supervisée et donne une vision globale des différents *clusters* de la distribution. En revanche elle est non linéaire et non reproductible donc difficilement interprétable. Elle dépend d'hyper-paramètres dont l'effet sur les projections est peu discuté dans les travaux sur les systèmes de traitement de la parole.

De telles projections permettent d'acquérir une intuition des phénomènes en jeu mais rarement de tirer des conclusions sur l'espace entier. Pourtant, dans la littérature, elles sont souvent utilisées pour discuter des propriétés des représentations. Parce qu'elle permet de rendre compte de la distribution des distances, la projection *t-SNE* est privilégiée dans les travaux récents. En reconnaissance de la parole dans [Pratap *et al.*, 2020], les représentations de la sortie de l'encodeur d'un modèle acoustique multilingue sont visualisées pour différentes voyelles. Des espaces d'*embeddings* sont visualisés en reconnaissance du locuteur avec la *t-SNE* [Cai *et al.*, 2020a]. En reconnaissance de la langue, l'*ACP* a été employée [Misra *et al.*, 2016], ainsi que la *LDA* [Misra *et al.*, 2016] et la *t-SNE* [Abdullah *et al.*, 2020, Shen *et al.*, 2018, Zhang et Hansen, 2017, Lin *et al.*, 2018]. Dans ces travaux, des observations sur la qualité des représentations et notamment sur la réduction de l'impact d'un facteur de variabilité (canal, genre, niveau de bruit) par rapport à une tâche de classification sont faites sur la base des visualisations à deux dimensions. Nous présentons ce type d'analyse dans la section suivante puis tentons de procéder à des mesures des phénomènes observés dans la suite de ce chapitre.

### 5.1.2 Configuration expérimentale

Conformément à la littérature récente, nous réalisons des projections *t-SNE* afin d'acquérir une intuition de la structure de l'espace des *embeddings*. Cela nous permet de préciser les propriétés des représentations nécessaires à une robustesse du système au changement de domaine. Les conclusions d'une telle analyse sont par nature limitées puisqu'elle porte sur des projections.

L'analyse est réalisée sur trois systèmes *x-vector* différents, dans le but de présenter un large panorama des structures possibles dans l'espace des représentations. Deux d'entre

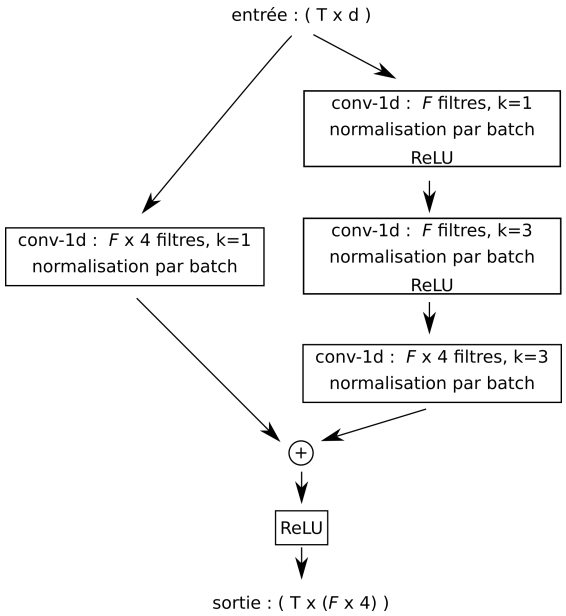



FIGURE 5.1 – Schéma d'un bloc résiduel utilisé dans le modèle *ResNet-1D*. $d$ est la dimension des vecteurs d'entrée et $T$ le nombre de trames d'entrée. $F$ est le nombre de filtres du bloc résiduel. Les couches $conv - 1d$ sont des convolutions selon la dimension temporelle avec un noyau dont la dimension est notée $k$, ces couches n'utilisent pas de paramètre de biais.

eux reposent sur une architecture résiduelle et sont entraînés sur différents canaux du corpus *RATS*, le dernier est basé sur un réseau *TDNN* et est entraîné sur le corpus *NIST LRE 2011*. Nous utilisons l'implémentation de l'algorithme de la bibliothèque *scikit-learn* [Pedregosa *et al.*, 2011].

**Systèmes *ResNet-1D* sur le corpus *RATS*** Les deux premiers systèmes sont entraînés sur le corpus *RATS* (annexe A.2), le premier uniquement sur le canal téléphonique (*src*) et le second sur tous les canaux. Nous réalisons les visualisations pour le canal téléphonique et pour un canal HF ($D$). Les deux systèmes partagent la même architecture et le même algorithme d'entraînement. Ils sont basés sur l'extraction d'*embeddings* d'un réseau *ResNet* avec des convolutions à une dimension (tableau 5.1). Les performances sont rapportées dans le tableau 5.3. Sans surprise, le système entraîné sur tous les canaux atteint une bonne performance pour les deux canaux de transmission testés alors que le système uniquement entraîné sur le canal téléphonique n'atteint une bonne performance que sur ce canal. De façon intéressante, le système entraîné sur tous les canaux est également plus performant sur le canal téléphonique. Pour ces systèmes, la dimension de l'espace des *x-vector* est 128.

TABLE 5.1 – Architecture du système *ResNet-1D*.

| Extraction de *bottleneck features* multilingues, calculés à partir de spectrogrammes en échelle *Mel*, puis production d'un *x-vector* par le réseau *ResNet-1D* entraîné pour l'identification de la langue, fourni en entrée d'un *SVM* qui prédit la langue. | | |
|---|---|---|
| Module | Modèle | Hyperparamètres |
| détection d'activité vocale | annotation manuelle | - |
| *features* acoustiques | spectrogrammes en échelle *Mel* | $\Delta_F = 25ms$, $\delta_F = 10ms$, 64 filtres |
| augmentations de données | aucune | - |
| *bottleneck features* | *bottleneck features* multilingues | Nous utilisons un modèle *Conformer* entraîné pour une tâche de reconnaissance de parole de bout en bout pour 14 langues du corpus *Babel*. Le modèle est raffiné au cours de l'entraînement du module d'identification de la langue selon la stratégie *3-étapes* exposée dans le chapitre 6. |
| modélisation du segment | architecture *ResNet-1D* (tableau 5.2) | entraîné pour l'identification de la langue avec l'entropie croisée, sélection du modèle final avec la méthode *stochastic weight averaging*, il est utilisé pour extraire des *embeddings* |
| classifieur final | *SVM* | LDA, centrage, blanchiment des *embeddings* et *SVM* |
| calibration | non utilisée | - |

TABLE 5.2 – Architecture du réseau *ResNet-1D*.
$n$ est le nombre de langues, $T_B$ est le nombre de trames d'entrée. Chaque série de blocs correspond à un enchaînement de blocs résiduels décrits en figure 5.1. Les *embeddings* sont extraits de la couche *fc1*.

| **Convolutions 1D selon l'axe temporel** | | | |
|---|---|---|---|
| nom de la série de blocs | blocs résiduels | filtres | dimension de sortie |
| blocs 1 | 3 | 16 | $T_B \times 64$ |
| blocs 2 | 4 | 32 | $T_B \times 128$ |
| blocs 3 | 6 | 64 | $T_B \times 256$ |
| blocs 4 | 3 | 128 | $T_B \times 512$ |
| ***Pooling* statistique : agrégation des moyennes et écarts-types** | | | |
| dimension de sortie : $2 \times 512 = 1024$ | | | |
| **Couches connectées** | | | |
| nom de la couche | dimension | | fonction d'activation |
| *fc1* | $1024 \times 128$ | | ReLU |
| *fc2* | $128 \times n$ | | Softmax |

TABLE 5.3 – Performance des systèmes *ResNet-1D* sur le corpus *RATS*, en fonction du canal d'entraînement du système. Moyenne des taux d'égale erreur (%) pour des segments de trois secondes.

| Canal d'entraînement | $EER_{avg}$ (%) | |
|---|---|---|
| | canal *src* | canal *D* |
| téléphone (*src*) | 6,24 | 33,54 |
| 9 canaux | **4,78** | **4,84** |

**Système *TDNN* sur le corpus *NIST LRE 2011*** Nous réalisons également la projection avec le système *x-vector bis* décrit dans le tableau 4.14 et entraîné sur le corpus *NIST LRE 2011* (annexe A.1.2), comprenant des enregistrements des canaux téléphoniques et *BNBS*. Le système reconnaît cinquante et une langues et est évalué pour vingt-quatre langues dont vingt sur les deux canaux. Nous utilisons le modèle entraîné avec l'entropie croisée et dont les performances sont rapportées dans le tableau 4.15. Pour ce système, la dimension de l'espace des *x-vector* est 512.

### 5.1.3 Analyse des représentations

Analysons maintenant les espaces de représentations de ces trois systèmes à l'aide de projections *t-SNE*.

**Systèmes *ResNet-1D* sur le corpus *RATS*** Les projections pour le système entraîné sur le canal téléphonique et pour le système entraîné sur tous les canaux sont représentées en figures 5.2 et 5.3, pour des enregistrements de trois secondes. Pour chaque

système, la projection est calculée pour les enregistrements des deux canaux, puis une figure est affichée pour chaque canal. Chaque enregistrement est représenté par un point, les couleurs correspondent aux langues et les symboles aux canaux de transmission.

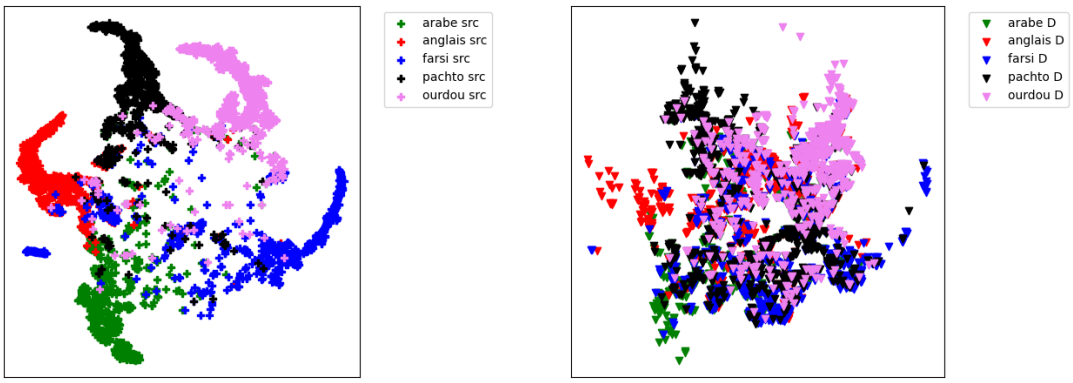
Pour le système entraîné sur le canal téléphonique, la figure 5.2a révèle que les représentations pour le canal téléphonique sont assez bien séparées en fonction des étiquettes de langue, en dépit de quelques mélanges. La figure 5.2b représente la même projection pour les enregistrements du canal HF *D*. Les groupes d'*embeddings* correspondant à chaque langue semblent rester approximativement à la même position que ceux du canal téléphonique. Cependant, les différentes langues sont très mélangées, et il est impossible de réaliser une classification efficace entre les langues dans cet espace à deux dimensions. Dans une telle situation, nous dirons que les représentations ne sont pas *discriminantes*, c'est-à-dire qu'elles ne permettent pas une bonne classification. Dans la suite de ce chapitre, nous développons des méthodes pour caractériser si des représentations dans l'espace des *x-vector* sont véritablement non *discriminantes*, sans nous limiter à une projection à deux dimensions.

À l'inverse, le système entraîné sur tous les canaux ne fait apparaître que de faibles différences entre les représentations des enregistrements téléphoniques (figure 5.3a) et des enregistrements du canal HF *D* (figure 5.3b). Non seulement les représentations sont discriminantes sur les deux canaux mais il semble que les enregistrements d'une même langue occupent la même zone dans l'espace des représentations, indépendamment du canal. C'est *a minima* le cas dans l'espace à deux dimensions de la projection. Cette situation est donc idéale. Aucune différence importante dans l'espace des représentations n'apparaît entre les deux canaux. Cela se traduit par une grande robustesse du système qui atteint des performances comparables sur les deux canaux (tableau 5.3). Nous dirons que les représentations sont *invariantes* selon le canal.

**Système *TDNN* sur le corpus *NIST LRE 2011*** Nous représentons une projection *t-SNE* de l'espace des *x-vector* du système *TDNN* pour les segments de dix secondes de l'ensemble de test du corpus *NIST LRE 2011* sur la figure 5.4. Nous utilisons la durée nominale dix secondes pour l'analyse de la variabilité en raison de la performance intermédiaire obtenue pour cette durée : les représentations sont assez discriminantes pour autoriser une bonne performance mais l'écart dû au canal peut impacter la performance. Les couleurs correspondent aux langues et les symboles aux canaux de transmission. Les deux canaux de transmission téléphone et *BNBS* (appels téléphoniques retransmis lors d'émissions de radio ou de télévision) sont représentés sur la même figure.

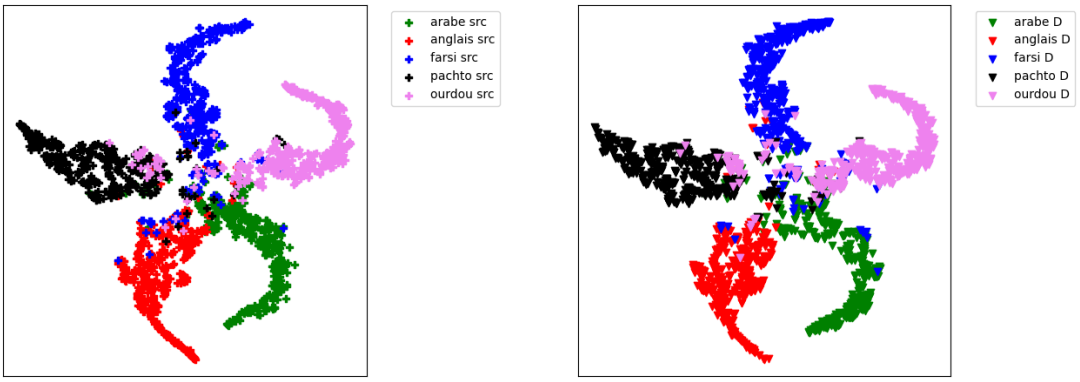
D'abord nous observons que les représentations sont bien regroupées par langue. De plus, la proximité entre les groupes de représentations associées à chaque langue semble cohérente avec les proximités linguistiques. On remarquera par exemple différents dialectes d'arabe en bleu ou plusieurs langues slaves en vert.

D'autre part, pour toutes les langues un décalage apparaît entre les deux canaux. Ce décalage ne semble pas impacter la capacité de discrimination avec les autres langues pour certaines langues, par exemple le mandarin, l'espagnol, le turc ou le tamoul, pour lesquelles l'écart entre les deux canaux semble faible par rapport à l'écart avec d'autres




(a) Canal téléphonique (*src*). (b) Canal HF (*D*).

FIGURE 5.2 – Projection *t-SNE* des *embeddings* de l'ensemble de test du corpus *RATS*, segments de trois secondes. Modèle entraîné sur le canal téléphonique (*src*). La projection est réalisée avec l'ensemble des deux canaux puis affichée séparément pour une plus grande lisibilité.




(a) Canal téléphonique (*src*). (b) Canal HF (*D*).

FIGURE 5.3 – Projection *t-SNE* des *embeddings* de l'ensemble de test du corpus *RATS*, segments de trois secondes. Modèle entraîné sur les neufs canaux du corpus. La projection est réalisée avec l'ensemble des deux canaux puis affichée séparément pour une plus grande lisibilité.

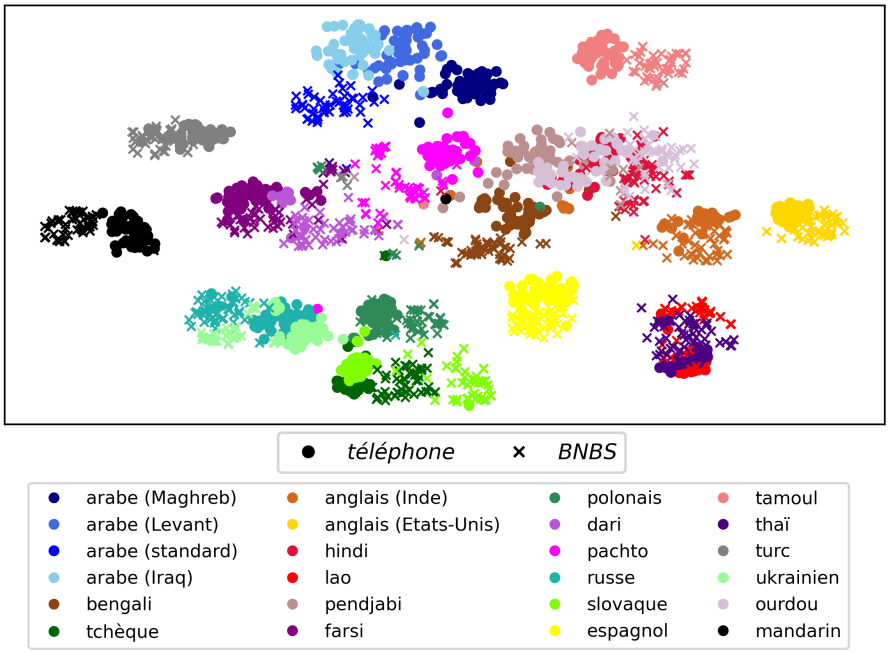



FIGURE 5.4 – Projection *t-SNE* des *embeddings* de l'ensemble de test du corpus *NIST LRE 2011*, segments de dix secondes. Les *embeddings* sont calculés avec le modèle entraîné avec l'entropie croisée.

langues. À l'inverse, pour certaines langues, l'écart entre les canaux semble entraîner une confusion avec d'autres langues proches, c'est le cas par exemple pour l'hindi et l'ourdou, le russe et l'ukrainien, ou le lao et le thaï. Pour ces paires de langues, les proximités entre *embeddings* semblent plus volontiers correspondre au canal qu'à la langue.

Pour mieux comprendre ces différentes configurations, nous réalisons plusieurs projections *t-SNE* de ces mêmes *x-vector* en nous limitant à des sous-ensembles des vingt-quatre langues du corpus de test (figure 5.5). Nous utilisons d'abord sept langues de familles différentes : anglais, arabe, mandarin, espagnol, hindi, russe et turc (figure 5.5a). Pour ces langues, l'espace des *x-vector* semble globalement séparé en zones correspondant à chaque langue et l'écart entre les représentations dû au canal ne semble pas impacter la capacité de discrimination, pour cette projection à deux dimensions. Cette configuration peut aussi être visualisée pour une unique paire de langue en figure 5.5b : l'anglais et le mandarin. Sur cette projection, il apparaît clairement qu'il existe un écart entre les canaux pour chaque langue, mais que celui-ci est négligeable devant l'écart entre les langues. La présence de deux domaines n'est donc pas un obstacle à la bonne classification.

Des configurations différentes apparaissent lorsqu'on s'intéresse à des paires de langues appartenant à la même famille. Pour deux dialectes de l'anglais (figure 5.5c), les écarts entre les langues et entre les canaux semblent être du même ordre de grandeur. Il n'y a cependant pas d'*incompatibilité* entre les deux distributions : il est possible d'apprendre une frontière entre les langues dans l'espace des *x-vector* qui demeurera valide pour les

deux canaux. C'est également le cas pour deux langues iraniennes (figure 5.5d), le farsi et le pachto. Pour les langues iraniennes, la configuration semble cependant moins favorable puisque la frontière entre les deux langues a une structure plus complexe. Cette comparaison nous permet d'introduire un troisième critère de qualité pour de bonnes représentations. En plus d'être *discriminantes* et *compatibles* entre les domaines, elles doivent *limiter la variabilité due au domaine* par rapport à la variabilité entre les langues afin d'éviter au classifieur un travail de compensation de l'effet du domaine. En effet une telle *compensation* signifie que le classifieur doit gérer la variabilité des représentations pour limiter l'impact du domaine, ce qui implique des modèles plus complexes d'une part, ainsi que l'obligation d'utiliser des données provenant des deux domaines lors de l'entraînement du classifieur.

### 5.1.4 Bilan des observations

L'analyse des visualisations en deux dimensions de l'espace des *x-vector* nous permet d'expliciter les propriétés attendues de représentations robustes.

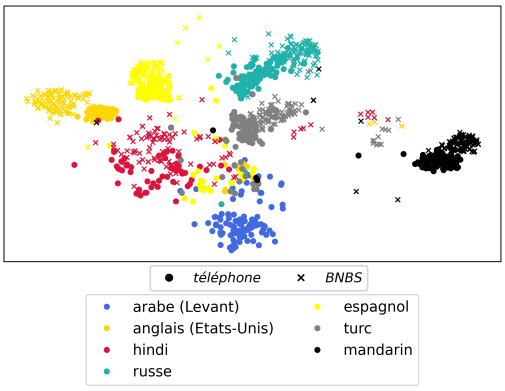
Elles doivent d'abord être *discriminantes* pour chacun des deux canaux, c'est-à-dire autoriser une classification ente les langues en séparant nettement les représentations associées à des langues différentes. Les figures 5.2a et 5.5a sont des exemples de représentations discriminantes, tandis que les figures 5.2b et 5.5e sont des exemples de représentations non discriminantes.

Les représentations doivent également être *compatibles* entre les canaux pour permettre d'entraîner un classifieur dans l'espace des représentations qui atteint une bonne performance sur chacun des canaux. Les représentations de la figure 5.2 ne sont pas compatibles entre les canaux, pour la simple raison qu'elles ne sont pas discriminantes pour le canal $D$. À l'inverse, les représentations des figures 5.5b et 5.5c sont compatibles. Bien que cela soit imaginable, nous n'avons jamais rencontré dans notre travail de représentations discriminantes et non compatibles.

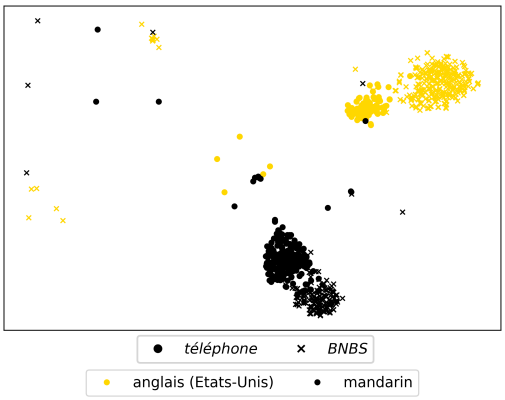
Un autre critère de qualité des représentations est la *limitation de la variabilité due au canal* par rapport à la variabilité due à la langue. Une telle propriété assure une stabilité de la position des représentations par rapport à la frontière entre les classes déterminée par un classifieur. Elle autorise donc l'entraînement d'un classifieur sur un seul canal. Les représentations de la figure 5.5b sont donc de qualité supérieures à celles de la figure 5.5c selon ce critère.

Enfin la configuration idéale correspond au cas où la variabilité des représentations due au canal peut être considérée comme nulle. Nous dirons alors que les représentations sont *alignées* entre les canaux ou *invariantes*. Les représentations de la figure 5.3 sont invariantes.

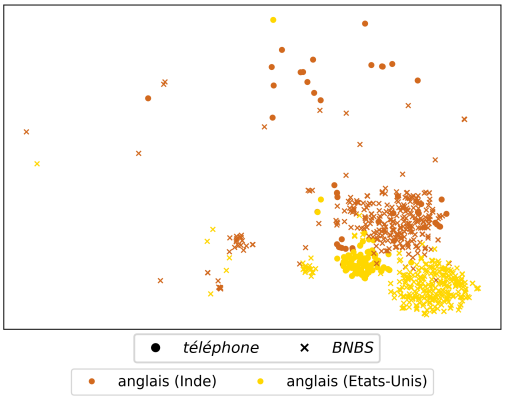
Il est cependant difficile de conclure sur la qualité des représentations à partir d'une visualisation en faible dimension. Dans la suite, nous tentons de caractériser de façon quantitative ces propriétés de l'espace des *embeddings*, dont nous avons donné des illustrations dans cette section.



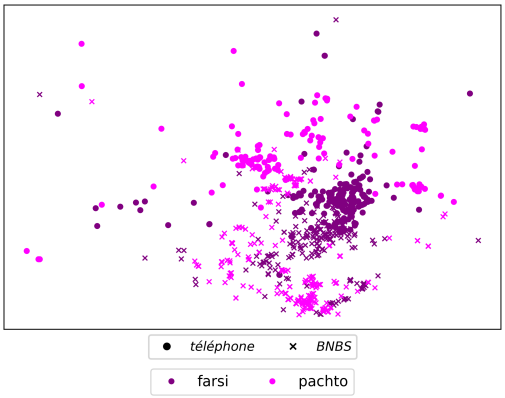

(a) Sept langues de familles différentes.



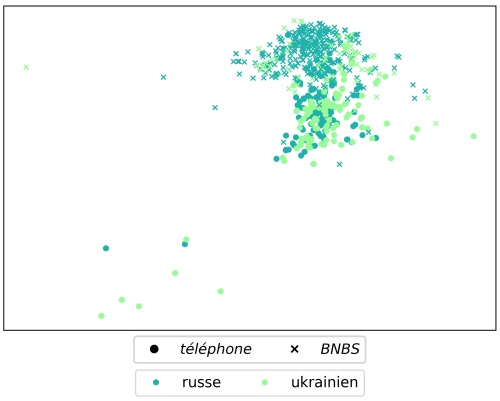

(b) Deux langues de familles différentes.



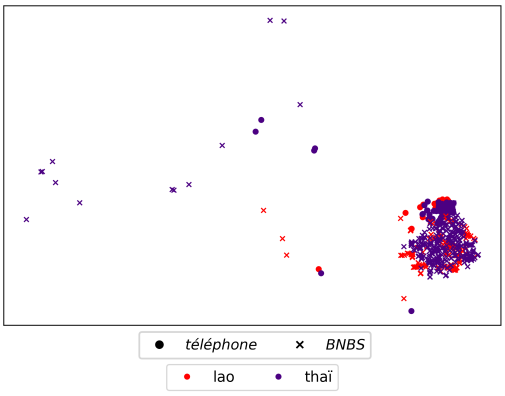

(c) Deux dialectes d'anglais.




(d) Deux langues iraniennes.




(e) Deux langues slaves.




(f) Deux langues taï.

FIGURE 5.5 – Projections *t-SNE* des *embeddings* de l'ensemble de test du corpus *NIST LRE 2011*, segments de dix secondes, pour certains sous-ensembles de langues. La totalité du corpus de test est représentée en figure 5.4. Ici une projection différente est réalisée pour chaque sous-ensemble de langues.

## 5.2 Caractérisation des représentations par la performance de classifieurs

Après avoir mis en lumière les propriétés attendues des représentations à partir de projections en deux dimensions, nous souhaitons caractériser quantitativement ces propriétés. Deux raisons motivent cette direction de travail. D'abord l'appréhension de propriétés géométriques dans des espaces de dimension élevée n'est pas intuitive et les structures que nous observons en deux dimensions dépendent étroitement des méthodes de projection choisies. D'autre part, les représentations présentent en général un certain degré de discrimination, de compatibilité, de limitation de la variabilité et d'invariance. Les méthodes que nous étudions ne visent pas seulement à transformer une configuration pathologique dans laquelle les représentations ne sont pas discriminantes en une configuration idéale dans laquelle elles sont parfaitement discriminantes et alignées. Elles doivent aussi permettre d'améliorer ces propriétés. Il est donc souhaitable de pouvoir mesurer cette évolution avec des critères quantitatifs. Une première approche est l'évaluation de la performance de plusieurs classifieurs entraînés sur des domaines différents à partir du même espace de représentations.

### 5.2.1 Notations

Dans le chapitre 2, nous avons introduit des notations pour chaque module du système de reconnaissance de la langue $s$.

$$s = Cal \circ Clas \circ R_e \circ B \circ F$$

où $F$ est l'extracteur de *features* acoustiques, $B$ l'extracteur de *bottleneck features*, $\mathcal{R}_e$ l'extracteur d'*embeddings*, $Clas$ le classfieur final et $Cal$ le module de calibration.

Dans le but d'évaluer la qualité de représentations, nous utilisons dans ce chapitre une vision plus abstraite :

$$s = clas \circ r$$

où $r$ est l'extracteur de représentations et $clas$ le classifieur. La séparation peut être faite au niveau de n'importe quel module du système. Dans ce travail, nous nous cantonnons à une étude de la qualité des *embeddings*. $clas$ et $r$ représentent donc les modules suivants.

$$clas = Cal \circ Clas \quad r = R_e \circ B \circ F$$

Nous proposons d'évaluer la qualité des représentations produites par $r$ à l'aune de la tâche de reconnaissance de la langue et de la robustesse à un changement de canal. Pour ce faire, nous employons le vocabulaire de l'adaptation de domaine. Soient donc un domaine source $\mathcal{D}_\mathcal{S}$ supposé correspondre au canal d'entraînement des représentations et un domaine cible $\mathcal{D}_\mathcal{T}$ potentiellement différent.

### 5.2.2 Méthode

Une manière d'évaluer la qualité des représentations produites par l'extracteur de représentations $r$ est l'entraînement de plusieurs classifieurs $clas$. Pour un extracteur de représentations $r$ fixé, on peut entraîner les classifieurs $clas_{\mathcal{S}}$, $clas_{\mathcal{T}}$ et $clas_{\mathcal{S}\cup\mathcal{T}}$ respectivement sur les domaines source, cible et sur l'union des deux domaines. En évaluant chacun des classifieurs sur les deux domaines, nous pouvons déterminer la qualité des représentations.

— Si $clas_{\mathcal{S}}$ atteint une bonne performance sur $\mathcal{D}_{\mathcal{S}}$ alors les représentations sont discriminantes sur le domaine source.

— Si $clas_{\mathcal{T}}$ atteint une bonne performance sur $\mathcal{D}_{\mathcal{T}}$ alors les représentations sont discriminantes sur le domaine cible.

— Si $clas_{\mathcal{S}\cup\mathcal{T}}$ atteint une bonne performance sur les domaines source et cible alors les représentations sur les deux domaines sont compatibles.

— Si $clas_{\mathcal{S}}$ atteint une bonne performance sur $\mathcal{D}_{\mathcal{T}}$ (ou de façon équivalente si $clas_{\mathcal{T}}$ atteint une bonne performance sur $\mathcal{D}_{\mathcal{S}}$) alors la variabilité des représentations au domaine ne fait pas obstacle à la classification.

Bien entendu les situations sont nuancées et on y répondra plus volontiers par le degré de performance d'un classifieur que par une réponse binaire.

### 5.2.3 Configuration expérimentale

Nous mettons en œuvre la méthode décrite pour les deux systèmes basés sur une architecture *ResNet-1D*, entraînés sur le corpus *RATS* dans le paragraphe 5.1.2. Les deux systèmes partagent la même architecture décrite dans le tableau 5.1 Nous évaluons deux types de représentations obtenues par entraînement mono-canal sur le canal téléphonique et entraînement sur les neuf canaux téléphonique et radios du corpus *RATS*. Par cohérence avec les projections réalisées dans la section précédente, le domaine source est le canal téléphonique (src) et le domaine cible est le canal HF *D*.

### 5.2.4 Performances

Pour chacun des deux extracteurs de représentations $r$, nous entraînons trois classifieurs : sur le domaine source, sur le domaine cible et sur les deux domaines. Nous évaluons chacun de ces classifieurs sur les deux domaines en termes de taux d'égale erreur moyen. Les résultats sont présentés dans le tableau 5.4.

Considérons d'abord les représentations entraînées sur le domaine source $\mathcal{D}_{\mathcal{S}}$. Le classifieur entraîné sur $\mathcal{D}_{\mathcal{S}}$ avec ces représentations atteint une bonne performance sur $\mathcal{D}_{\mathcal{S}}$ ($EER_{avg} = 6{,}24\%$). C'est donc que les représentations sont discriminantes sur $\mathcal{D}_{\mathcal{S}}$. À l'inverse, le classifieur entraîné sur $\mathcal{D}_{\mathcal{S}}$ atteint une performance très faible sur le domaine cible $\mathcal{D}_{\mathcal{T}}$ (33,54%). Cela signifie que les représentations ne sont pas invariantes.

Le classifieur entraîné sur $\mathcal{D}_{\mathcal{T}}$ avec les représentations entraînées sur $\mathcal{D}_{\mathcal{S}}$ n'atteint pas une bonne performance sur $\mathcal{D}_{\mathcal{T}}$ (24,97%). Les représentations ne sont donc pas discriminantes sur $\mathcal{D}_{\mathcal{T}}$. La dégradation de la performance sur $\mathcal{D}_{\mathcal{S}}$ (12,21%) par rapport au

TABLE 5.4 – Performance de classifieurs entraînés pour cinq langues sur le corpus *RATS*, en fonction des méthodes d'entraînement de l'extracteur de représentations $r$ et du classifieur *clas*. Moyenne des taux d'égale erreur pour des segments de trois secondes.

| | Canal d'entraînement de $r$ | | | |
|---|---|---|---|---|
| | sur le domaine source ($\mathcal{D}_\mathcal{S}$) | | sur neuf canaux | |
| Canal d'entraînement de *clas* | $EER_{avg}$ (%) sur $\mathcal{D}_\mathcal{S}$ | sur $\mathcal{D}_\mathcal{T}$ | $EER_{avg}$ (%) sur $\mathcal{D}_\mathcal{S}$ | sur $\mathcal{D}_\mathcal{T}$ |
| source ($\mathcal{D}_\mathcal{S}$, canal *src*) | **6,24** | 33,54 | 4,99 | 5,05 |
| cible ($\mathcal{D}_\mathcal{T}$, canal *D*) | 12,21 | **24,97** | 4,92 | **4,75** |
| source et cible | 6,98 | 25,47 | **4,78** | 4,84 |

classifieur entraîné sur $\mathcal{D}_\mathcal{S}$ (6,24%) est nette mais pas totale. Cela semble indiquer que le problème principal des représentations n'est pas la compatibilité entre les canaux mais bien la discriminabilité sur le domaine cible.

Ce constat est confirmé par la performance du classifieur entraîné sur les deux canaux avec ces représentations entraînées sur $\mathcal{D}_\mathcal{S}$. Il atteint une performance sur les deux canaux comparable à la performance sur chacun des canaux d'un classifieur entraîné sur le canal correspondant (6,98% et 25,47%). Il n'y a pas de problème de compatibilité.

Les représentations entraînées sur tous les canaux permettent d'atteindre des performances similaires sur les deux canaux et indépendamment du canal d'entraînement du classifieur (entre 4,75% et 5,05%). La variabilité des représentations due au canal a peu d'effet sur la performance de classification. Ceci est un indice en faveur de l'invariance de ces représentations entraînées sur neuf canaux.

### 5.2.5 Conclusion sur la performance des classifieurs

L'entraînement et l'évaluation de classifieurs sur des domaines individuels et sur des ensembles de domaines permet d'évaluer la discriminabilité des représentations, leur compatibilité, ainsi que l'effet de la variabilité due au domaine sur la classification. Ce type d'approche présente cependant deux inconvénients. L'entraînement de plusieurs classifieurs présente une lourdeur, dépendant de la complexité de l'algorithme d'entraînement ; il nécessite *a minima* la définition d'ensembles d'entraînement et de test pour chaque domaine. D'autre part les conclusions obtenues par l'approche basée sur les classifieurs ne sont par définition valides que pour les architectures de classifieurs utilisées. Pour cette raison, nous souhaitons directement caractériser les représentations, indépendamment des classifieurs qui seront ensuite entraînés. En effet des représentations de bonne qualité ont vocation à pouvoir être traitées par une grande diversité de classifieurs, comme les *bottleneck features* de *BUT / Phonexia* [Silnova *et al.*, 2018] ou bien les *i-vectors* du *NIST i-vector machine learning challenge 2015* [Tong *et al.*, 2016].

## 5.3 Caractérisation directe des espaces de représentation

Nous avons observé sur les visualisations un écart entre les distributions associées aux différents canaux de transmission dans l'espace des représentations. Nous proposons de mesurer cette variabilité de deux manières : avec le rapport des covariances inter-classes et intra-classes et avec des divergences entre groupes d'*embeddings*.

### 5.3.1 Rapports de covariance

La variabilité d'une distribution peut être estimée en premier lieu par sa matrice de covariance. Lorsque la distribution peut être partagée en plusieurs classes, la covariance peut être divisée en covariance intra-classes et covariance inter-classes. Nous introduisons donc des nouvelles notations afin d'englober sous le même point de vue la partition en classes correspondant aux langues, ou la partition en classes correspondant aux domaines. Notons $C$ l'ensemble fini des classes. Pour chaque classe $c \in C$, supposons disposer de $n_c$ échantillons $\{x_i^c\}_{i=1}^{n_c}$ correspondant à cette classe.

Dans les formules qui suivent, $\Sigma$, la matrice de covariance globale de la distribution, est calculée en équilibrant les différentes classes. Elle peut donc être partagée en covariance intra-classes $\Sigma_{\text{intra}}$ et covariance inter-classes $\Sigma_{\text{inter}}$.

$$\Sigma = \Sigma_{\text{intra}} + \Sigma_{\text{inter}}$$

où des estimateurs des matrices de covariance sont donnés par :

$$\forall c \in \mathcal{C}, \quad \mu_c = \frac{1}{n_c} \sum_{i=1}^{n_c} x_i^c \quad \text{et} \quad \Sigma_c = \frac{1}{n_c} \sum_{i=1}^{n_c} (x_i^c - \mu_c)(x_i^c - \mu_c)^T$$

$$\mu = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} \mu_c \quad \Sigma = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} \frac{1}{n_c} \sum_{i=1}^{n_c} (x_i^c - \mu)(x_i^c - \mu)^T$$

$$\Sigma_{\text{intra}} = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} \Sigma_c \quad \text{et} \quad \Sigma_{\text{inter}} = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} (\mu_c - \mu)(\mu_c - \mu)^T$$

Afin de rendre compte de l'importance de la division en classes $\mathcal{C}$ dans la variabilité de la distribution, nous utilisons le rapport $\mathcal{R}$ entre les traces des matrices de covariance inter-classes et intra-classes. Les matrices de covariance étant symétriques semi-définies positives, leurs traces sont positives. Le rapport varie donc entre 0 et $+\infty$. Il correspond à la part de la variance de la distribution pouvant être attribuée aux classes $\mathcal{C}$. Ce rapport sera nul si les classes ont la même moyenne $\mu_c$ et très important si la majorité de la variabilité correspond à la séparation en classes.

$$\mathcal{R} = \frac{\mathcal{T}r(\Sigma_{\text{inter}})}{\mathcal{T}r(\Sigma_{\text{intra}})}$$

où $\mathcal{T}r$ est l'opérateur calculant la trace d'une matrice, somme des valeurs propres.

En fonction de la division en classes choisie, cet outil peut mettre en lumière plusieurs effets. Ainsi nous pouvons mesurer la discriminabilité entre les langues ou entre les canaux. Remarquons que ce rapport correspond au critère utilisé pour déterminer les paramètres de la *LDA* [Tharwat *et al.*, 2017]. L'algorithme consiste à co-diagonaliser les deux matrices de covariance intra-classes et inter-classes et à sélectionner les vecteurs propres qui maximisent le rapport de covariance. Dans le cas où les classes représentent les langues, le rapport entre les covariances inter-classes et intra-classes est donc une mesure de la qualité des représentations cohérente avec l'usage qui en est fait dans les systèmes de reconnaissance de la langue. Les représentations sont en effet souvent fournies à un classifieur final dont le premier traitement est une *LDA*.

### 5.3.2 Mesure de divergence

Les rapports entre les matrices de covariance permettent seulement de rendre compte de l'impact global d'un facteur de variabilité (celui qui est utilisé pour définir les classes $\mathcal{C}$). Afin d'aller plus loin dans l'analyse, il est souhaitable de mesurer directement la différence entre des groupes d'*embeddings*. Pour cela n'importe quelle mesure de divergence peut être utilisée. Dans notre travail, nous utilisons la *MMD*, afin d'être cohérent avec la mesure de divergence appliquée pour la régularisation des réseaux de neurones. Notons $d$ la mesure de divergence choisie et $X_1$ et $X_2$ des groupes d'*embeddings*. Alors on mesurera simplement la divergence $d(X_1,X_2)$. Une telle approche a été appliquée en utilisant la *MMD* pour la reconnaissance d'images [Haeusser *et al.*, 2017].

En définissant différents groupes d'*embeddings* en fonction des langues et des domaines (les canaux dans notre cas), nous pourrons comparer la variabilité due aux domaines à la variabilité entre les langues.

## 5.4 Analyse des représentations de systèmes d'identification de la langue

Nous montrons maintenant l'intérêt des deux types de mesures définies dans la section précédente pour évaluer la robustesse de représentations de reconnaissance de la langue à des facteurs de variabilité.

### 5.4.1 Motivation de l'analyse

Pour des systèmes de reconnaissance de la langue, nous souhaitons évaluer la qualité des représentations produites par le réseau de neurones d'identification de la langue, en terme de variabilité due au canal par rapport à la variabilité due à la langue. Nous mesurons la variabilité des représentations d'abord pour des modèles entraînés par apprentissage supervisé, avec un ou plusieurs canaux, puis pour des modèles entraînés avec les méthodes de régularisation exposées dans le chapitre 4. Cela nous permet de comprendre l'effet de ces méthodes sur l'espace des représentations.

### 5.4.2 Configuration expérimentale

**Systèmes analysés** Nous considérons deux systèmes : un système *CNN* (tableau 4.8) et un système *x-vector* appelé *x-vector bis* (tableau 4.14).

**Corpus utilisés** Les expériences avec le systèmes *CNN* sont réalisées sur le corpus *OpenSAD 15* (annexe A.2) avec quatre langues : anglais, arabe, ourdou et pachto. Certains modèles analysés correspondent à des expériences d'adaptation de domaine non supervisée. Dans ce cas, le domaine source est le canal $G$ et le domaine cible le canal $D$. Cette configuration expérimentale correspond aux expériences préliminaires du chapitre 4 qui nous ont permis de fixer les hyper-paramètres des différentes méthodes.

Pour le système *x-vector*, nous utilisons le corpus *NIST LRE 2011* (annexe A.1.2), comprenant des enregistrements téléphoniques et des données *BNBS*. Le système est entraîné avec cinquante et une langues et les canaux téléphonique et *BNBS*, puis évalué pour vingt-quatre langues dont vingt sont présentes sur les deux canaux.

### 5.4.3 Rapports de covariance

Nous analysons les représentations produites par un système de type *CNN* sur le corpus *OpenSAD 15*. Plusieurs systèmes sont entraînés. D'abord des entraînements supervisés sont réalisés, sur les canaux $G$, puis $D$, puis les sept canaux du corpus. Nous appliquons ensuite la méthode d'alignement parallèle entre les canaux $G$ et $D$ sur différentes couches décrite en section 4.2.5. Enfin nous réalisons des expériences d'adaptation de domaine non supervisée avec les méthodes de régularisation développées dans la section 4.3.5. Afin de permettre leur analyse, les méthodes d'adaptation non supervisées sont appliquées sur la couche de *pooling* en plus de la couche de sortie, en dépit de leur faible performance. Les performances de reconnaissance de la langue de chaque système sur les canaux $G$ et $D$ sont rapportées en terme de moyenne des taux d'égale erreur ($EER_{avg}$).

Pour chaque système, nous mesurons plusieurs rapports de covariance en faisant varier la couche du réseau utilisée pour extraire les représentations et les classes utilisées pour définir les matrices de covariance. Cela nous permet de mesurer à la fois la variabilité due au canal et à la langue, ainsi que d'observer l'évolution de ces variabilités entre les couches du réseau. Deux couches du réseau *CNN* sont utilisées pour calculer des rapports de covariance (nous suivons les notations du tableau 4.7) : la couche de sortie *fc2* qui est effectivement utilisée pour réaliser les prédictions et la couche de *pooling* qui correspond à la première couche produisant une représentation pour l'ensemble du segment. Trois rapports de covariance sont mesurés. La discriminabilité entre les langues sur le canal $G$ est notée $\mathcal{R}_G^L$. Pour ce calcul, seuls les enregistrements du canal $G$ sont utilisés et les classes correspondent aux quatre langues du corpus *OpenSAD 15*. De façon identique, la discriminabilité des langues sur le canal $D$ est notée $\mathcal{R}_D^L$. Enfin la discriminabilité des canaux est notée $\mathcal{R}^{\mathcal{C}}$. Elle est calculée avec toutes les données, la division en classes correspondant aux canaux. Les mesures de rapport de covariance sont réalisées sur l'ensemble de test du corpus et présentées dans le tableau 5.5.

TABLE 5.5 – Analyse du système *CNN* entraîné sur le corpus *OpenSAD 15* avec plusieurs méthodes d'entraînement : apprentissage supervisé, alignement parallèle entre les canaux $G$ et $D$ et adaptation de domaine non supervisée du canal $G$ vers le canal $D$. La performance est mesurée sur les canaux $G$ et $D$. Des rapports de covariance sont mesurés sur la couche de sortie (*fc2*) et sur la couche de *pooling*.

| Méthode d'entraînement | | *pooling* | *fc2* | | | $EER_{avg}$ (%) | |
|---|---|---|---|---|---|---|---|
| couche | régularisation | $\mathcal{R}^{\mathcal{C}}$ | $\mathcal{R}^{\mathcal{C}}$ | $\mathcal{R}^{L}_{G}$ | $\mathcal{R}^{L}_{D}$ | $G$ | $D$ |
| Apprentissage supervisé | | | | | | | |
| sur le canal $G$ | | 0,28 | 0,13 | 0,39 | **0,12** | 14 | 48 |
| sur le canal $D$ | | 0,94 | 0,07 | **0,05** | 0,60 | 53 | 15 |
| sur tous les canaux | | **0,43** | $\mathbf{3.10^{-4}}$ | 0,43 | 0,55 | 15 | 12 |
| Apprentissage supervisé sur $G$ avec alignement parallèle avec le canal D | | | | | | | |
| *pooling* | alignement parallèle | $\mathbf{5.10^{-3}}$ | $6.10^{-3}$ | 0,50 | 0,32 | 11 | 29 |
| *fc1* | | 0,07 | $1.10^{-3}$ | 0,56 | 0,41 | 11 | 21 |
| *fc2* | | 0,18 | $1.10^{-3}$ | 0,46 | 0,44 | 13 | **17** |
| Adaptation non supervisée de $G$ vers $D$ | | | | | | | |
| pooling | distance moyennes | $\mathbf{6.10^{-3}}$ | $\mathbf{5.10^{-3}}$ | 0,53 | 0,09 | 11 | 44 |
| | MMD | 0,02 | 0,02 | 0,52 | 0,09 | 10 | 45 |
| | CORAL | 0,27 | 0,09 | 0,39 | 0,12 | 14 | 43 |
| *fc2* | distance moyennes | 0,58 | $8.10^{-3}$ | 0,54 | 0,10 | 12 | 39 |
| | MMD | **0,02** | $\mathbf{4.10^{-4}}$ | 0,83 | **1,09** | 11 | **9** |
| | CORAL | **0,01** | $\mathbf{6.10^{-4}}$ | 0,54 | **0,67** | 11 | **10** |

Ces mesures permettent d'abord de retrouver des effets attendus. Pour les systèmes entraînés de façon supervisée sur un seul canal, la discriminabilité entre les langues sur la couche de sortie est plus forte pour le canal d'entraînement (0,39 et 0,60) que pour l'autre canal (0,12 et 0,05) et la différence entre les canaux est assez importante sur cette couche (0,13 et 0,07). Les rapports de covariance confirment simplement l'analyse qui peut être réalisée avec la moyenne des taux d'égale erreur. En revanche le système entraîné de façon supervisée sur l'ensemble des canaux atteint une bonne discriminabilité entre les langues pour les deux canaux (0,43 et 0,55) et un écart faible entre les canaux sur la couche de sortie ($3.10^{-4}$). Remarquons que pour ces trois systèmes la discriminabilité entre les canaux est très importante sur la couche de *pooling* (0,28, 0,94 et 0,43). C'est donc que le système fonctionnant sur tous les canaux utilise des représentations intermédiaires dépendant du canal et qu'il ne fusionne les représentations des deux domaines que dans l'espace des prédictions.

Pour les systèmes entraînés avec alignement parallèle, nous remarquons que l'augmentation de la discriminabilité entre les langues sur le canal $D$ (0,32, 0,41 puis 0,44) est associée à une diminution de la discriminabilité entre les canaux sur la couche de sortie ($6.10^{-3}$, $1.10^{-3}$ puis $1.10^{-3}$). De façon intéressante, la diminution drastique de l'écart entre les domaines sur la couche de *pooling* ($5.10^{-3}$ lorsque l'alignement parallèle est

appliqué sur cette couche) induit une diminution de la discriminabilité entre les canaux sur la couche de sortie ($6.10^{-3}$) mais qui ne se traduit pas pas une augmentation très importante de la performance sur le domaine $D$ (taux d'égale erreur moyen de 29%).

Cela explique pourquoi l'adaptation non supervisée fonctionne sur la couche de sortie (taux d'égale erreur moyens de 39%, 9% et 10%) et pas sur la couche de *pooling* (44%, 45% et 43%). Les régularisations sur la couche de *pooling* réduisent bien l'écart entre les canaux sur cette couche ($6.10^{-3}$, 0,02 et 0,27) et par conséquent sur la couche de sortie ($5.10^{-3}$, 0,02 et 0,09) mais n'améliorent pas la discriminabilité entre les langues (0,09, 0,09 et 0,12). À l'inverse, les méthodes de régularisation sur la couche de sortie permettent, tout en réduisant la discriminabilité entre canaux ($8.10^{-3}$, $4.10^{-4}$ et $4.10^{-4}$) d'augmenter la discriminabilité entre les langues pour les deux canaux.

Les méthodes de régularisation sur la couche de sortie qui permettent de réaliser une adaptation efficace (*MMD* et *CORAL*) réduisent la discriminabilité entre les canaux sur la couche de sortie. Elles réduisent également fortement la discriminabilité entre les canaux sur la couche de *pooling*. Ce n'est pas le cas pour le système entraîné de façon supervisée sur plusieurs canaux qui préserve une séparation importante entre les canaux sur la couche de *pooling*.

Il y a donc une différence majeure entre un système entraîné sur plusieurs canaux et un système entraîné avec une contrainte d'invariance non supervisée entre les canaux. Un système entraîné de façon supervisée sur plusieurs canaux préserve des représentations différenciées en fonction du canal sur ces couches intermédiaires mais apprend une fonction de classification qui fonctionne pour l'ensemble de ces représentations. À l'inverse, une régularisation visant à l'alignement des domaines sur la couche de sortie impose un alignement des domaines sur les couches précédentes et donc une représentation plus invariante au domaine. Les représentations sont donc de meilleure qualité, au sens où la variabilité due au domaine n'a pas à être compensée par le classifieur. Cela peut expliquer l'amélioration de la performance sur le domaine source obtenue par régularisation du réseau.

### 5.4.4 Mesure de divergence

Nous étudions le système *x-vector bis* sur le corpus *NIST LRE 2011*. Les divergences sont mesurées pour l'ensemble de test de durée nominale dix secondes. Cinq systèmes correspondant à la même architecture sont analysés, ils correspondent à l'utilisation des fonctions de coût suivantes lors de leur entraînement : entropie croisée (*CE*), entropie croisée et *MMD*, entropie croisée et *n-pair loss*, *additive angular magin softmax* (*AAM*), *additive angular magin softmax* et *n-pair loss*. Les performances de ces systèmes sont rapportées dans le tableau 4.15. Afin d'être cohérents avec la méthode de régularisation utilisée, nous utilisons comme divergence $d$ pour caractériser l'écart entre groupes d'*embeddings* la *MMD* avec le noyau $k_{\text{énergie}}(x_1,x_2) = -||x_1 - x_2||_2$.

**Impact du canal**

Nous mesurons d'abord cette divergence pour des paires de groupes d'*embeddings* définies par la langue et le canal. Pour une langue $L$ et un domaine $D$, on notera $X^{L,D}$ l'ensemble des *embeddings* correspondant à cette langue et ce domaine. Il semble cependant délicat de tirer des conclusions de l'observation des divergences entre toutes les paires d'ensembles d'*embeddings* correspondant à tous les canaux et langues du corpus. Nous proposons donc d'agréger les divergences afin de disposer de quelques métriques qui rendent compte de la variabilité des représentations imputables à la langue et au domaine. Ainsi, pour une langue $L$ donnée et deux domaines $D_1$ et $D_2$, il est intéressant de mesurer la variabilité entre les deux domaines des *embeddings* associés.

$$\mathcal{D}_{D_1,D_2}(L) = d(X^{L,D_1},X^{L,D_2})$$

À l'inverse, pour un domaine $D$, on peut mesurer l'écart des représentations entre deux langues $L_1$ et $L_2$.

$$\mathcal{D}_{L_1,L_2}(D) = d(X^{L_1,D},X^{L_2,D})$$

Nous souhaitons maximiser cette divergence afin d'obtenir les représentations les plus discriminantes possibles sur ce domaine. Pour une langue $L$, nous mesurons donc la discriminabilité avec la langue engendrant la plus grande confusion.

$$\mathcal{D}_L(D,L) = \underset{L' \neq L}{min}\, \mathcal{D}_{L,L'}(D)$$

Notons $T$ le canal téléphonique et $B$ le canal *BNBS*. Pour chaque langue, nous mesurons donc trois quantités : la divergence entre les canaux $\mathcal{D}_{T,B}(L)$, la divergence avec la langue la plus proche pour le canal téléphonique $\mathcal{D}_L(T,L)$ et pour le canal *BNBS* $\mathcal{D}_L(B,L)$. Nous moyennons ces trois quantités pour l'ensemble des langues et caractérisons ainsi les variabilités dues au canal et la discriminabilité entre les langues pour chaque canal.

En raison des propriétés du corpus d'entraînement du système, nous moyennons ces valeurs par groupe de langue : langues observées sur les deux canaux lors de l'apprentissage $G_A$, langues observées uniquement sur le canal téléphonique $G_T$, langues observées uniquement sur le canal *BNBS* $G_B$. Les compositions des groupes de langues sont rappelées dans le tableau A.4. De plus, afin de permettre une comparaison facile entre différents systèmes, les divergences sont normalisées pour chaque système en les divisant par la divergence moyenne entre les langues sur le canal téléphonique pour le groupe de langues $G_A$. Nous affichons ces valeurs moyennées et normalisées des divergences pour les cinq systèmes en figure 5.6.

Pour le système entraîné avec l'entropie croisée, la divergence entre les canaux est aussi forte que la divergence entre les langues pour les langues observées sur les deux canaux et plus forte pour les langues observées uniquement sur un canal. Nous en tirons deux constats. D'abord, la variabilité au canal existe dans les représentations même pour un système entraîné sur les deux domaines, ce qui confirme l'observation réalisée avec les rapports de covariance pour un système entraîné sur plusieurs canaux. D'autre part la variabilité au canal est plus forte pour des langues observées uniquement sur un canal.

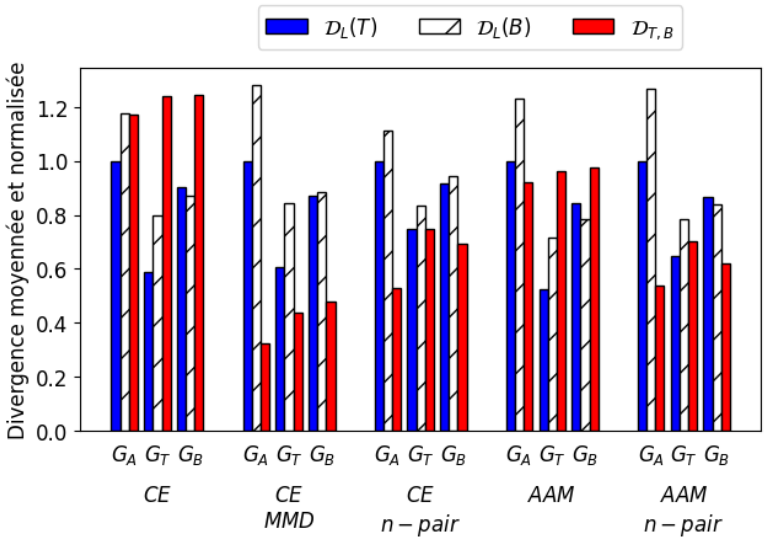



FIGURE 5.6 – Variabilité due au canal de transmission. Moyenne des divergences entre groupes d'*embeddings* pour cinq modèles et trois groupes de langues. Ensemble de test du corpus *NIST LRE 2011*, segments de dix secondes. Trois groupes de langues sont utilisés : langues observées sur les deux canaux ($G_A$), langues uniquement observées sur le canal téléphonique ($G_T$) et langues uniquement observées sur le canal *BNBS* ($G_B$), voir tableau A.4. Les divergences sont normalisées par système en fonction de la discriminabilité entre les langues sur le canal téléphonique pour le groupe de langues $G_A$.

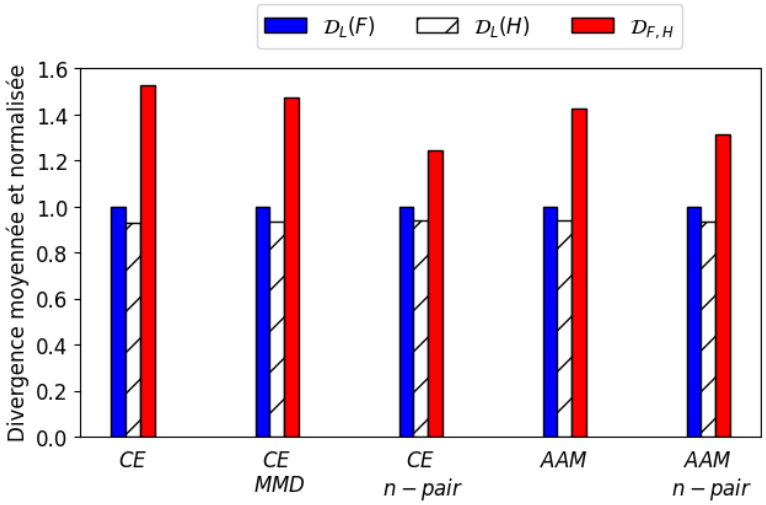



FIGURE 5.7 – Variabilité due au genre. Moyenne des divergences entre groupes d'*embeddings* pour cinq modèles. Ensemble de test du corpus *NIST LRE 2011*, segments de dix secondes. Les divergences sont moyennées par système en fonction de la divergence entre langues pour les femmes.

Nous calculons les mêmes divergences moyennées pour les quatre modèles entraînés avec d'autres fonctions de coût : entropie croisée et *MMD*, entropie croisée et *n-pair loss*, *AAM*, et enfin *AAM* et *n-pair loss*. D'abord le modèle *AAM* souffre du même problème de variabilité que le modèle entraîné avec l'entropie croisée, dans une moindre mesure. Ensuite la régularisation avec la *MMD* a très fortement réduit la variabilité due au canal, y compris pour les langues uniquement observées sur un canal lors de l'apprentissage. Enfin, la régularisation avec la *n-pair loss* réduit également la variabilité due au canal, de façon moins efficace que la *MMD* mais sans utiliser d'annotation de domaine.

### Impact du genre

La même analyse peut être effectuée pour d'autres facteurs de variabilité. Pour l'illustrer, nous nous intéressons au genre pour les mêmes systèmes. Nous notons donc, pour une langue $L$, $X^{L,F}$ et $X^{L,H}$ les *embeddings* respectivement associés aux femmes et aux hommes et mesurons les divergences suivantes.

$$\mathcal{D}_{F,H}(L) = d(X^{L,F}, X^{L,H})$$

$$\mathcal{D}_L(F,L) = min_{L' \neq L} \mathcal{D}_{L,L'}(F) \quad \mathcal{D}_L(H,L) = min_{L' \neq L} \mathcal{D}_{L,L'}(H)$$

Les deux genres étant observés pour toutes les langues dans l'ensemble d'apprentissage, nous moyennons les divergences sur l'ensemble des langues. Nous les normalisons en fonction de la divergence entre les langues pour les femmes. Les divergences moyennées et normalisées sont représentées sur la figure 5.7.

D'abord nous observons que la variabilité due au genre est très importante comparée à la discriminabilité entre les langues. Sans surprise, la *MMD* appliquée aux canaux ne réduit pas cette variabilité. En revanche, la *n-pair loss* la réduit de façon certaine. Cela confirme notre hypothèse, la *n-pair loss* réduit de façon implicite des variabilités observées dans l'ensemble d'apprentissage, sans utiliser d'annotation de ces variabilités.

### 5.4.5 Conclusion sur l'analyse d'un système de reconnaissance de la langue

La mesure de rapports de covariance et de divergences dans un espace de représentations permet donc de mettre en lumière le comportement des modèles vis-à-vis de facteurs de variabilité. Un facteur de variabilité (canal ou genre) se traduit par un décalage dans l'espace des représentations, du même ordre de grandeur que l'écart entre les langues, même lorsque les langues ont été observées sur les deux domaines au cours de l'apprentissage. Les deux fonctions de régularisation étudiées, *MMD* et *n-pair loss* limitent cette variabilité mais obéissent à des logiques différentes. La *MMD* réduit très fortement une variabilité de façon supervisée. À l'inverse la *n-pair loss* n'utilise pas d'étiquettes de domaine (canal ou genre) et réduit donc les variabilités rencontrées dans l'ensemble d'entraînement de façon non supervisée. Les régularisations engendrent un espace de représentations unifié, ce qui produit des systèmes de natures différentes d'un apprentissage multi-domaines pour lequel les représentations varient en fonction du domaine mais où la variabilité est compensée par le classifieur.

## 5.5 Conclusion

Dans ce chapitre nous avons montré que la robustesse d'un système à un facteur de variabilité peut être mesurée dans un espace d'*embeddings*. Nous avons d'abord visualisé l'espace des *x-vector* à l'aide de projections à deux dimensions, ce qui nous a permis de définir les propriétés des représentations nécessaires à la robustesse au changement de domaine. Nous avons ensuite montré que la simple mesure de la performance des systèmes en entraînant plusieurs classifieurs sur des ensembles différents permettait de rendre compte de certaines de ces propriétés. Ce genre d'évaluation peut cependant dépendre du type de classifieur utilisé. Nous avons donc justifié l'idée que les représentations devaient être directement évaluées. Nous avons introduit deux outils pour caractériser l'espace des *embeddings* : le rapport entre covariances inter-classes et intra-classes, ainsi que la mesure de divergences entre groupes d'*embeddings*. Nous avons montré l'intérêt de ces outils en les mettant en œuvre pour caractériser la robustesse d'un système de reconnaissance de la langue à des facteurs de variabilité. Ces outils sont génériques et peuvent être mobilisés pour analyser la robustesse d'un système pour une autre tâche, l'annexe E montre l'application de ces outils pour un système de reconnaissance automatique de la parole.

Ces analyses de l'espace des *embeddings* nous ont permis d'acquérir une compréhension plus profonde des systèmes que la simple mesure de performance. Nous avons

ainsi pu mettre en lumière plusieurs phénomènes concernant le processus d'entraînement d'un modèle de reconnaissance de la langue. Lors d'un apprentissage multi-domaines, les représentations intermédiaires restent séparées et les domaines ne sont rassemblés que sur les couches proches de la couche de sortie. C'est comme si on avait plusieurs modèles parallèles qui traitent chaque domaine. De plus les représentations d'un système de reconnaissance de la langue sont moins discriminantes pour les domaines non observés lors de l'apprentissage. Les régularisations par alignement des domaines permettent d'augmenter la discriminabilité sur le domaine cible ainsi que de diminuer la variabilité due au canal par rapport à l'écart entre les langues. Contrairement à l'apprentissage multi-domaines, les régularisations par alignement augmentent l'invariance au domaine des couches intermédiaires, tendant vers un modèle unifié qui fonctionne sur tous les domaines. La régularisation par *metric learning* réduit des variabilités inconnues *a priori*.

# 6

# Étude de l'extracteur de *bottleneck features*

Dans les deux chapitres précédents, nous avons discuté de méthodes d'augmentation et d'analyse de la robustesse au canal de transmission qui opèrent dans un espace de représentation à l'échelle du segment audio. Cependant, pour les systèmes à l'état de l'art, ces représentations de l'ensemble du segment sont produites à partir de vecteurs caractérisant le signal à l'échelle d'une trame, appelés *bottleneck features*. Ces *bottleneck features* sont extraits par un réseau de neurones entraîné pour une tâche de reconnaissance de la parole. Nous nous intéressons maintenant à ce module.

Nous positionnons d'abord notre travail dans la littérature récente. Les *bottleneck features* couramment utilisés sont extraits de modèles multilingues entraînés pour une tâche d'identification de phones à l'échelle d'une trame. Deux dynamiques sont à l'œuvre dans le but d'améliorer ce modèle standard.

D'abord, les développements récents de modèles de reconnaissance de la parole avec des architectures dites *de bout en bout* sont appliqués à l'extraction de *bottleneck features*. Cela permet de bénéficier de nouvelles architectures et de fonctions de coût dites de séquence à séquence par opposition à une tâche de classification par trame. Des premiers travaux ont montré l'intérêt de ce type d'approches. Nos contributions consistent à montrer que, de même que les *bottleneck features* classiques, ces modèles peuvent être entraînés à partir de corpus multilingues et qu'un gain important peut être obtenu par une régularisation de leur entraînement.

D'autre part, la question de la meilleure stratégie d'entraînement d'un système constitué d'un extracteur de *bottleneck features* et d'un modèle d'identification de la langue est largement ouverte. Des travaux ont montré l'intérêt d'une stratégie d'apprentissage conjointe des deux modules pour des architectures classiques de reconnaissance de phones à l'échelle d'une trame. Nous comparons les stratégies d'apprentissage avec un nouveau système obéissant au paradigme de séquence à séquence et montrons l'apport d'un apprentissage conjoint.

## 6.1 Positionnement par rapport à la littérature récente

Positionnons d'abord notre travail par rapport à la littérature sur les *bottleneck features*. Rappelons que le principe d'extraction de *bottleneck features* a été décrit en section 2.4.3. Dans ce chapitre, nous présentons d'abord l'analyse qui a été faite des systèmes *bottleneck features* de référence du point de vue de la robustesse au canal de transmission. Ensuite, nous décrivons les travaux existants concernant les deux directions de nos contributions : l'utilisation de recettes de reconnaissance de la parole *de bout en bout* et le développement d'une stratégie d'entraînement conjointe du module d'extraction de *bottleneck features* et du module d'identification de la langue.

### 6.1.1 Différentes propriétés des *features* en fonction de l'architecture et de la recette d'entraînement

La tâche de reconnaissance de la parole est largement utilisée pour entraîner des systèmes d'extraction de *bottleneck features* à l'échelle de la trame [Jiang *et al.*, 2014, McLaren *et al.*, 2016b]. Un système de reconnaissance de la parole est utilisé pour réaliser un alignement forcé entre des trames acoustiques et des étiquettes de phones. Un réseau de neurones est ensuite entraîné à prédire l'étiquette de phone associée à chaque trame. Les activations d'une couche intermédiaire d'un tel réseau peuvent être utilisées comme une représentation de la trame, elles sont appelées *bottleneck features*. Ce type de représentations est utilisé dans les systèmes de reconnaissance de la langue à l'état de l'art [Fér *et al.*, 2017, Snyder *et al.*, 2018].

Les *bottleneck features* portent des informations permettant de réaliser la tâche ayant servi à leur entraînement. Cependant, l'objectif final n'étant pas la tâche de reconnaissance de la parole mais la reconnaissance de la langue, comment sélectionner le meilleur système de *bottleneck features* ? La méthode d'évaluation se révèle assez coûteuse. Il faut évaluer les *bottleneck features* en fonction de la performance de reconnaissance de la langue qu'ils autorisent, ce qui suppose d'entraîner un système de reconnaissance de la langue avec ces *features*. Deux observations majeures ont été faites dans le cadre de telles études.

D'abord, la meilleure performance de reconnaissance de la parole n'est pas associée à la meilleure performance de reconnaissance de la langue [Lozano-Diez *et al.*, 2017]. Cela n'est pas surprenant : les deux tâches, bien que complémentaires sont différentes. C'est cependant assez problématique pour ces systèmes où l'apprentissage est réalisé en deux phases avec d'abord l'entraînement des *bottleneck features* dont le modèle est sélectionné en fonction de la performance de reconnaissance de la parole, puis l'entraînement du système de reconnaissance de la langue.

D'autre part, des propriétés différentes peuvent être attendues en fonction de la position de la couche *bottleneck* dans le réseau *bottleneck features*. En particulier, la position optimale de la couche *bottleneck* est proche de la couche de sortie lorsque les conditions de test correspondent aux conditions d'entraînement du réseau *bottleneck features* alors qu'une couche plus centrale est préférable en cas de changement de canal de transmission, comme observé pour le corpus *RATS* [McLaren *et al.*, 2016b].

### 6.1.2 Modèles dits *de bout en bout*

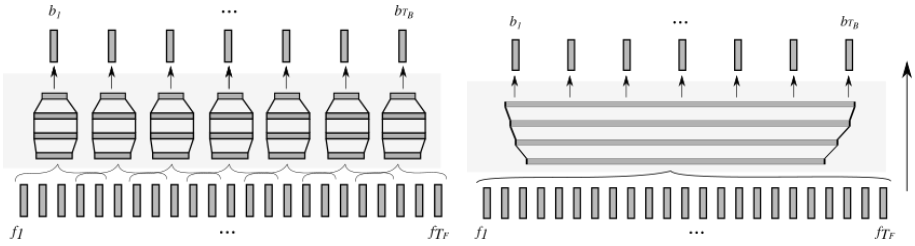
Classiquement, l'alignement forcé entre les trames acoustiques et les étiquettes de phones est réalisé par un système de reconnaissance de la parole, reposant par exemple sur des chaînes de Markov cachées (*Hidden Markov Model - HMM*) ou sur une approche hybride incluant une chaîne de Markov cachée et des réseaux de neurones profonds. Ensuite, un réseau de neurones est entraîné pour prédire les étiquettes de phones à l'échelle de la trame et des *bottleneck features* peuvent être extraits de ce réseau. Depuis le milieu des années 2010, des systèmes de reconnaissance de la parole de bout en bout (*end-to-end* en anglais) ont été introduits [Gulati *et al.*, 2020, Amodei *et al.*, 2016]. Il s'agit de réseaux de neurones profonds directement entraînés à produire une transcription du signal d'entrée. Ils prennent en entrée une séquence de *features* acoustiques et produisent un posteriogramme sur les étiquettes de sortie. Ils peuvent être entraînés avec une fonction de coût dite *de séquence à séquence*, comme la *connectionist temporal classification loss* (*CTC*) [Graves et Jaitly, 2014]. Cela signifie que l'ensemble de la séquence d'entrée est traité pour produire une séquence de sortie, contrairement aux modèles de *bottleneck features* classiques qui fonctionnent à l'échelle de la trame. Ces modèles permettent d'extraire des représentations d'une couche intermédiaire et donc de produire directement des *bottleneck features* du modèle de reconnaissance de la parole *de bout en bout* sans besoin de recourir à un alignement forcé et à l'entraînement d'un nouveau réseau. Cela a deux avantages. D'abord la recette d'entraînement du système s'en trouve grandement simplifiée. Ensuite, la question de l'exactitude des prédictions de phones réalisées par le modèle de reconnaissance de la parole est rendue caduque.

L'extraction de *bottleneck features* à partir d'un modèle de reconnaissance de la parole *de bout en bout* a permis d'obtenir une performance de reconnaissance de la langue à l'état de l'art sur le corpus *NIST LRE 2007*, en utilisant une architecture *Transformer* entraînée uniquement sur de l'anglais [Ling *et al.*, 2020]. D'autre part, une comparaison intéressante a été effectuée pour une tâche de reconnaissance de dialectes chinois [Ren *et al.*, 2019]. Elle a montré que l'utilisation de l'alignement forcé pour entraîner un modèle de *bottleneck features* ne permettait pas d'améliorer la performance par rapport à l'utilisation directe d'*embeddings* extraits du réseau *de bout en bout*.

Les architectures de *bottleneck features* classiques sont radicalement différentes des modèles de séquence à séquence, comme représenté sur la figure 6.1. Les premières appliquent le même traitement à chaque trame tandis que les secondes traitent l'ensemble de la séquence pour produire une séquence de *bottleneck features* de sortie.

### 6.1.3 Stratégies d'entraînement

L'entraînement du système de reconnaissance de la langue est réalisé de façon séquentielle. D'abord l'extracteur de *bottleneck features* est entraîné pour une tâche de reconnaissance de la parole, il est ensuite utilisé pour extraire des *features* et le modèle d'identification de la langue est entraîné. Comme nous l'avons cependant déjà mentionné, la meilleure performance en terme de reconnaissance de phones n'assure pas que le modèle permet d'extraire les meilleures représentations pour la reconnaissance de la



(a) Architecture classique de *bottleneck features* trame à trame

(b) Modèle de séquence à séquence

FIGURE 6.1 – Représentation de l'extraction de *bottleneck features* pour une séquence de trames acoustiques.

langue [Lozano-Diez *et al.*, 2017]. Il apparaît même qu'une meilleure performance de reconnaissance de la parole peut être atteinte avec des représentations des phones indépendantes des langues, ce qui ouvre la voie à un entraînement antagoniste à la langue du modèle de reconnaissance de la parole [Adams *et al.*, 2019]. Il n'y a pas aujourd'hui de méthode pour sélectionner le meilleur compromis entre les tâches d'identification de la langue et de reconnaissance de la parole pendant l'entraînement d'un système de reconnaissance de la langue.

Une première étape pour répondre à cette question consiste à inclure les deux modules dans une unique architecture dite *de bout en bout*. Plusieurs stratégies d'apprentissage deviennent alors possibles. La première est l'entraînement *de bout en bout* des deux modules avec la tâche d'identification de la langue sans tenir compte de l'information de reconnaissance de la parole. Elle permet uniquement de bénéficier d'une architecture plus profonde pour la tâche d'identification de la langue sans utiliser l'information phonétique supplémentaire. Une autre stratégie est un *fine-tuning* (raffinement) des poids du réseau *bottleneck features* en rétro-propageant la fonction de coût d'identification de la langue, après que celui-ci ait été entraîné de façon classique avec une tâche d'identification de phones. D'après [Tang *et al.*, 2017b], ces deux stratégies d'apprentissage ne permettent pas d'améliorer la performance de reconnaissance de la langue.

Une autre approche est l'apprentissage conjoint (*multi-task*) des deux modules avec les deux fonctions de coût. Elle a été mise en œuvre avec succès en utilisant une [Zhao *et al.*, 2019, Li *et al.*, 2021] et deux [Tang *et al.*, 2017b] langues pour la tâche de reconnaissance de la parole. Ces travaux ne proposent cependant pas de méthode automatique pour choisir les poids respectifs des deux tâches lors de l'entraînement.

Les travaux sur les stratégies d'entraînement discutés dans cette section étudient le paradigme classique de *bottleneck features* entraînés pour une tâche de classification à l'échelle d'une trame. Notons l'exception de [Punjabi *et al.*, 2021] qui propose une stratégie d'entraînement conjointe pour l'anglais et l'hindi avec une architecture de reconnaissance de la parole *de bout en bout*.

## 6.2 Modèle de reconnaissance de la parole de bout en bout multilingue

### 6.2.1 Idée

Des modèles de reconnaissance de la parole *de bout en bout* ont été utilisés pour extraire des *bottleneck features*. Ils n'ont cependant été entraînés que pour une seule langue, l'anglais [Ling *et al.*, 2020] ou le mandarin [Ren *et al.*, 2019]. Nous avons développé un modèle multilingue de bout en bout pour le calcul de *bottleneck features*. D'autre part nous utilisons l'architecture *Conformer* [Gulati *et al.*, 2020] introduite en 2020, par opposition à un modèle *Transformer* pour [Ling *et al.*, 2020]. Ce modèle d'extraction de *bottleneck features* a été développé dans le cadre de notre participation à la compétition *Oriental Language Recognition 2020*, puis évalué avec le corpus *NIST LRE 2007*.

### 6.2.2 Description du modèle de reconnaissance de la parole multilingue

Pour réaliser un modèle de reconnaissance de la parole multilingue, nous nous inspirons des modèles multilingues de *bottleneck features* classiques. La méthode la plus couramment utilisée consiste en un encodeur partagé suivi d'une couche de sortie spécifique à chaque langue [Fér *et al.*, 2017].

#### Principe de l'architecture

Nous utilisons l'architecture *Conformer* [Gulati *et al.*, 2020] qui a donné des résultats de reconnaissance de la parole à l'état de l'art sur le corpus *librispeech*. Cette architecture vise à combiner les couches d'auto-attention qui permettent de modéliser des dépendances à long terme avec des couches de convolution selon la dimension temporelle qui permettent de modéliser des dépendances locales. Elle est fondée sur une structure élémentaire appelée bloc *Conformer*, correspondant à une alternance de couches d'auto-attention et de convolution. Un réseau *Conformer* est constitué d'un encodeur, lui-même formé d'un enchaînement de blocs *Conformer*, et d'un décodeur. Sur le modèle des architectures de *bottleneck features* classiques, nous définissons un modèle *Conformer* multilingue avec un encodeur partagé et un décodeur pour chaque langue.

Ce modèle prend en entrée des spectrogrammes avec des trames échantillonnées toutes les dix millisecondes. L'encodeur multilingue est constitué d'une couche convolutive de sous-échantillonnage produisant des trames échantillonnées toutes les quarante millisecondes, ainsi que d'une série de blocs *Conformer* [Gulati *et al.*, 2020]. Les hyperparamètres permettant de caractériser l'encodeur sont le nombre de blocs *Conformer*, leur nombre de filtres, le nombre de têtes d'attention utilisées et la taille des noyaux de convolution.

Un décodeur spécifique est défini pour chaque langue utilisée pour l'entraînement du réseau. Nous utilisons des décodeurs simples constitués d'une seule couche linéaire suivie d'un *softmax*. Les *bottleneck features* correspondent aux activations de la couche de sortie de l'encodeur du modèle *Conformer*.

#### Méthode d'entraînement

Dans toutes nos expériences, nous entraînons le modèle de reconnaissance de la parole avec la *CTC loss*. Ce choix, conforme aux autres travaux existant dans la littérature, provient du rôle de l'extracteur de *bottleneck features* au sein du système de reconnaissance de la langue. Il s'agit de produire une représentation du signal à l'échelle d'une trame et non pas de modéliser des dépendances dans l'ensemble du segment. La *CTC loss* qui repose sur une hypothèse d'indépendance des prédictions pour des trames successives conditionnellement aux représentations latentes utilisées s'accorde avec l'idée de produire la meilleure représentation de chaque trame. De plus, afin de ne pas biaiser l'apprentissage en raison des proportions des différentes langues dans l'ensemble d'entraînement, nous équilibrons chaque *minibatch* selon les langues. La fonction de coût utilisée pour l'entraînement est donc la moyenne arithmétique des *CTC loss* pour l'ensemble des langues visées.

Enfin il convient de choisir un alphabet de sortie pour le modèle de reconnaissance de la parole. Plusieurs options existent dans la littérature : des graphèmes, des phones, des morceaux de phrase (*sentence pieces*). Les phones permettent de concentrer l'apprentissage sur l'information phonétique, ils nécessitent cependant de disposer d'un dictionnaire de transcription phonétique pour les langues traitées. Les graphèmes sont un choix naturel. Cependant le nombre de graphèmes varie dans des proportions importantes entre les langues alphabétiques et celles basées sur des idéogrammes, ce qui pose le problème de l'équilibre entre les différentes fonctions de coût lorsqu'un modèle est entraîné avec ces deux types de langues. C'est pourquoi nous avons décidé d'entraîner un modèle de *sentence piece*. Cette approche consiste à définir un ensemble de séquences de caractères qui permettent de recouvrir un ensemble de textes d'entraînement (souvent les transcriptions de l'ensemble d'entraînement du modèle de reconnaissance de la parole). Elle ne nécessite aucune connaissance de la langue traitée et permet de fixer *a priori* le nombre d'étiquettes utilisées. Nous avons utilisé le modèle de *sentence piece open source* des auteurs de [Kudo et Richardson, 2018].

#### Détails d'implémentation

Nous utilisons un petit modèle *Conformer* avec deux blocs *Conformer* et quatre têtes d'auto-attention avec 80 ou 64 canaux et un noyau de convolution de dimension 17. Le nombre de canaux correspond à la dimension des *bottleneck features* et est choisi afin d'être comparable aux autres *features* utilisés pour l'entraînement de modèles d'identification de la langue : des bancs de filtres de dimension 64 et les *bottleneck features* de *BUT/Phonexia* [Silnova *et al.*, 2018] de dimension 80.

### 6.2.3 Performance

Nous évaluons cette méthode d'entraînement de *bottleneck features* en la mettant en œuvre dans deux cadres expérimentaux : le corpus *OLR 2020* et le corpus *NIST LRE 2007*.

### *OLR 2020*

Dans le cadre de la compétition *OLR 2020*, nous avons entraîné un extracteur de *bottleneck features*. Les règles du *challenge* [Li *et al.*, 2020a] interdisaient l'utilisation d'autres corpus que ceux définis par la compétition et donc l'utilisation de modèles pré-entraînés sur ceux-ci. Nous nous limitons donc à une comparaison avec des *features* spectrales.

**Corpus** Les corpus utilisés sont décrits en annexe A.3. Le système *bottleneck features* est entraîné sur l'ensemble des enregistrements des corpus de la compétition fournis avec des transcriptions, soit les corpus *AP16-OL7* et *AP17-OL3*. Ce sont des enregistrements téléphoniques de dix langues : cantonais, coréen, indonésien, japonais, kazakh, mandarin, ouïghour, russe, tibétain et vietnamien.

Le modèle de reconnaissance de la langue est entraîné avec des enregistrements de canal téléphonique pour les six langues pour lesquelles nous disposons de canaux inconnus dans le corpus de la compétition *OLR 2020* : japonais, mandarin, ouïghour, russe, tibétain et vietnamien. Cela nous permet d'évaluer le système pour trois ensembles de test : *test-2018* qui correspond à un canal téléphonique, *dev-2019* avec des enregistrements de canaux inconnus et *test-2019*, mélange de canal téléphonique et de canaux inconnus. Ces corpus correspondent à ceux utilisés pour la phase appelée *sélection d'un système robuste* et sont décrits en annexe A.3.

**Architecture de l'extracteur de *bottleneck features*** Pour cette série d'expériences, nous utilisons des *bottleneck features* de dimension 64, afin de les comparer avec les bancs de filtres en échelle *Mel* que nous utilisons, de même dimension.

**Système de reconnaissance de la langue** Le système de reconnaissance de la langue est présenté dans le tableau 6.1. Le modèle d'identification de la langue est le *TDNN* classique [Snyder *et al.*, 2018]. Les entrées du modèle sont les *bottleneck features* extraits du modèle de reconnaissance de la parole de type *Conformer*. Nous évaluons directement les sorties du réseau de neurones sans classifieur final.

**Résultats** Les résultats en terme de performance de reconnaissance de la langue de trois systèmes entraînés avec différents *features* sont rapportés dans le tableau 6.2. La moyenne des taux d'égale erreur ($EER_{avg}$) est utilisée afin de limiter l'impact de la calibration et de nous concentrer sur l'évaluation des *features*.

Nous observons d'abord que, pour tous les ensembles de test, la performance est toujours meilleure sur le canal téléphonique (*test-2018*) que sur les canaux inconnus. De plus, les *MFCC* permettent d'atteindre une meilleure performance que les bancs de filtres en échelle *Mel* sur le canal téléphonique ($EER_{avg}$ de 4,75% au lieu de 11,95%) mais sont moins robustes au changement de canal, comme le montre la performance sur l'ensemble *dev-2019* (37,78% au lieu de 25,48%). Pour tous les ensembles de test, les *bottleneck*

TABLE 6.1 – Architecture du système reposant sur un *TDNN*

| Détection de la parole, extraction de *features* acoustiques, puis de *bottleneck features*, fournis en entrée d'un réseau d'identification de la langue produisant une probabilité *a posteriori* par langue. | | |
|---|---|---|
| Module | Modèle | Hyperparamètres |
| détection d'activité vocale | système basé sur l'énergie du signal | - |
| *features* acoustiques | spectrogrammes en échelle *Mel* ou *MFCC* | $\Delta_F = 25$ ms, $\delta_F = 10$ ms, 64 filtres *Mel*, 24 coefficients *MFCC*, *CMVN* |
| augmentations de données | quatre augmentations | *specAugment* [Park *et al.*, 2019], ajout de bruit blanc, et de *babble noise*, filtres passe-bande aléatoires |
| *bottleneck features* | réseau *Conformer* (ou non utilisés) [Gulati *et al.*, 2020] | détails exposés dans ce chapitre. |
| modélisation du segment | *TDNN* standard [Snyder *et al.*, 2018] | entraîné par descente de gradient stochastique pour l'identification de six langues avec l'entropie croisée, les augmentations de données et la stratégie *stochastic weight averaging* [Izmailov *et al.*, 2018] |
| classifieur final | non utilisé | utilisation directe des probabilités *a posteriori* produites par le réseau |
| calibration | non utilisée | évaluation en terme d'$EER_{avg}$ |

*features* extraits du modèle de reconnaissance de la parole multilingue surpassent les *features* spectrales. Ce sont donc des représentations à la fois plus pertinentes en conditions connues et plus robustes au canal de transmission.

TABLE 6.2 – Performance de reconnaissance de la langue sur les ensembles de test du corpus *OLR 2020* en terme de moyenne des taux d'égale erreur ($EER_{avg}$, %).

| *features* | test-2018 | dev-2019 | test-2019 |
|---|---|---|---|
| spectrogramme en échelle *Mel* | 11,95 | 25,48 | 29,54 |
| *MFCC* | 4,75 | 37,78 | 24,34 |
| *bottleneck features* | **3,91** | **19,11** | **11,08** |

**Régularisation des *bottleneck features*** Rappelons que notre travail vise à l'augmentation de la robustesse au canal de transmission. Dans les chapitres précédents, nous nous sommes concentrés sur le modèle d'identification de la langue dans le but de produire des représentations de l'ensemble du segment robustes au changement de canal. La même idée peut être appliquée aux *bottleneck features*. Dans le but d'augmenter leur robustesse, nous leur avons appliqué plusieurs méthodes de régularisation : une normalisation cepstrale des moyennes et des variances (*CMVN - cepstral mean and variance normalization*), *specAugment* [Park *et al.*, 2019] et les trois augmentations de données décrites dans cette section.

Les performances sont d'abord évaluées en terme de reconnaissance de la parole sur le corpus de validation. Nous mesurons la performance simplement à l'aide de la fonction de coût ayant servi à l'entraînement du système : la moyenne des *CTC loss* pour les dix langues cibles. Nous entraînons quatre modèles différents en ajoutant à chaque fois une nouvelle méthode de régularisation. Le dernier modèle utilise donc toutes les méthodes de régularisation. Les résultats sont rapportés dans le tableau 6.3. Ils révèlent que chaque méthode de régularisation permet une augmentation de la performance de reconnaissance de la parole, alors même que le système est évalué sur un corpus téléphonique correspondant aux mêmes conditions que le corpus d'entraînement.

TABLE 6.3 – Performance de reconnaissance automatique de la parole en fonction de la recette d'entraînement du modèle de reconnaissance de la parole. La performance est mesurée en terme de *CTC loss* moyenne.

| Nom des *features* | Recette d'entraînement | *CTC loss* sur l'ensemble de validation |
|---|---|---|
| *BNF de base* | spectrogrammes *Mel* | 3,94 |
| | + *CMVN* | 3,69 |
| | + *specAugment* | 3,52 |
| *BNF finaux* | + augmentations de données | **3,02** |

Nous extrayons donc des *bottleneck features* du modèle de reconnaissance de la parole entraîné avec toutes les méthodes de régularisation. Nous les nommons *BNF finaux*,

TABLE 6.4 – Performance de reconnaissance de la langue sur les ensembles de test du corpus *OLR 2020* en terme de $EER_{avg}$ (%).

| *features* | test-2018 | dev-2019 | test-2019 |
|---|---|---|---|
| *BNF de base* | 3,91 | 19,11 | **11,08** |
| *BNF finaux* | **3,43** | **17,24** | 13,68 |

par opposition aux *BNF de base*, entraînés sans régularisation et déjà évalués. Nous entraînons ensuite un modèle d'identification de la langue en utilisant les *BNF finaux*. La performance est rapportée dans le tableau 6.4. Nous observons une amélioration de la performance de reconnaissance de la langue, tant sur le canal téléphonique (*test-2018*), $EER_{avg}$ de 3,43% au lieu de 3,91%, que sur les canaux inconnus (*dev-2019*), 17,24% au lieu de 19,11%. Une légère dégradation est observée sur le corpus contenant à la fois des données téléphoniques et de canaux inconnus (*test-2019*), 13,68% au lieu de 11,08%. Nous ne concluons donc pas sur la robustesse des *bottleneck features* régularisés mais constatons qu'ils permettent une meilleure performance de reconnaissance de la langue en conditions connues.

### *NIST LRE 2007*

Les expériences avec le corpus *OLR 2020* ont permis d'établir l'intérêt des *bottleneck features* extraits d'un modèle de reconnaissance de la parole *de bout en bout*. Nous comparons maintenant ces nouveaux *bottleneck features* aux *bottleneck features* classiques.

**Corpus** Nous comparons notre approche au modèle de *bottleneck features* pré-entraînés de *BUT/Phonexia* [Silnova *et al.*, 2018]. Ce modèle est entraîné sur dix-sept langues du corpus *Babel* : assamais, bengali, cantonais, cebuano, créole haïtien, kazakh, kurmandji, lao, lituanien, pachto, tagalog, tamoul, télougou, tok pisin, turc, vietnamien, zoulou. Pour entraîner nos propres modèles, nous nous sommes limités à quatorze langues pour la simple raison que nous ne disposons pas des corpus cantonais, cebuano et tagalog.

Afin d'évaluer uniquement la performance de reconnaissance de la langue, nous nous plaçons dans des conditions connues et idéales. Nous utilisons la tâche de reconnaissance de quatorze langues de l'évaluation *NIST LRE 2007*. Les corpus de test contiennent uniquement des enregistrements téléphoniques peu bruités et les langues cibles sont très distinctes, contrairement à des évaluations postérieures qui introduisaient des classifications parmi des langues ou dialectes voisins. Les quatorze langues cibles sont les suivantes : allemand, anglais, arabe, bengali, chinois, coréen, espagnol, farsi, hindoustani, japonais, russe, tamoul, thaï et vietnamien. Trois ensembles de test sont fournis correspondant à trois durées nominales : trois, dix et trente secondes. Le modèle d'identification de la langue est entraîné uniquement sur ces quatorze langues et sur un canal téléphonique avec les corpus *NIST LRE 2003*, *2005*, *2007 train*, *2009*, *Callfriend* et *NIST SRE 2008*. Cet ensemble d'entraînement est le même que pour des publications récentes utilisant le même corpus d'évaluation [Cai *et al.*, 2020b, Ling *et al.*, 2020], il est décrit en annexe A.1.2.

**Architecture de l'extracteur de *bottleneck features*** Pour cette série d'expériences, nous utilisons des *bottleneck features* de dimension 80, afin de les comparer avec les *bottleneck features* pré-entraînés de *BUT/Phonexia* [Silnova *et al.*, 2018], de même dimension. Le modèle de reconnaissance de la parole est entraîné avec les régularisations *CMVN* et *specAugment*. Nous n'utilisons pas d'augmentation de données.

**Système de reconnaissance de la langue** Le système de reconnaissance de la langue est décrit dans le tableau 6.5. Pour cette série d'expériences, le modèle d'identification de la langue est le *TDNN* classique [Snyder *et al.*, 2018]. Les entrées du modèle sont les *bottleneck features* extraits du modèle *Conformer*. Le modèle d'identification de la langue est entraîné pour minimiser l'entropie croisée avec la régularisation *specAugment* et aucune augmentation de données. Nous sélectionnons le modèle avec la stratégie *stochastic weight averaging* [Izmailov *et al.*, 2018]. De plus le modèle est entraîné avec des segments de trois secondes. Un modèle est raffiné (*fine-tuned*) pour chacune des autres durées de test (dix et trente secondes). Il n'y a pas de classifieur final, nous utilisons directement les prédictions du réseau d'identification de la langue. Les paramètres de calibration sont appris sur un ensemble de validation pour chaque durée de test.

**Résultats** Trois modèles d'identification de la langue sont entraînés avec différents *features* : des bancs de filtres en échelle *Mel*, les *bottleneck features* classiques de *BUT/Phonexia* et les *bottleneck features* extraits du modèle de reconnaissance de la parole *Conformer*. Les performances de reconnaissance de la langue sont rapportées dans le tableau 6.6, avec les métriques standards de la communauté dans le but d'autoriser une comparaison avec la littérature : $C_{avg}$, $minDCF$ et $EER$.

Pour toutes les durées, les deux types de *bottleneck features* permettent un gain de performance très important par rapport aux bancs de filtres *Mel*. Le modèle *Conformer* permet de produire des *features* de qualité équivalente à l'approche classique. Ils sont légèrement moins performants pour des segments courts (trois et dix secondes) et meilleurs pour des segments longs de trente secondes, avec un $C_{avg} \times 100$ de 0,77 au lieu de 1,37.

### 6.2.4 Conclusion sur les *bottleneck features* extraits d'un modèle de reconnaissance de la parole *de bout en bout*

À l'issue de ces premières comparaisons expérimentales, nous avons montré qu'un modèle de reconnaissance de la parole *de bout en bout* peut être utilisé pour extraire des *bottleneck features*. Ces *bottleneck features* permettent une meilleure performance de reconnaissance de la langue que des *features* spectrales, et une performance du même ordre que des *bottleneck features* classiques entraînés avec une tâche de reconnaissance de phones. De plus, cette approche présente l'avantage d'une recette d'entraînement simplifiée, qui ne repose pas sur un alignement forcé entre représentations spectrales et transcriptions. Enfin, l'augmentation de données au cours de l'entraînement du modèle de reconnaissance de la parole *de bout en bout* peut augmenter la performance de généralisation des *bottleneck features*.

TABLE 6.5 – Architecture du système de reconnaissance de la langue pour le corpus *NIST LRE 2007*

| Détection de la parole, extraction de *features* acoustiques, puis de *bottleneck features*, fournis en entrée d'un réseau d'identification de la langue produisant une probabilité *a posteriori* par langue, régression logistique. | | |
|---|---|---|
| Module | Modèle | Hyperparamètres |
| détection d'activité vocale | système basé sur l'énergie du signal | - |
| *features* acoustiques | spectrogrammes en échelle *Mel* | $\Delta_F = 25$ ms, $\delta_F = 10$ ms, 64 filtres *Mel*, normalisation de la moyenne et de la variance |
| augmentations de données | *specAugment* [Park *et al.*, 2019] uniquement | - |
| *bottleneck features* | réseau *Conformer* [Gulati *et al.*, 2020] (ou *bottleneck features* pré-entraînés de *BUT/Phonexia* [Silnova *et al.*, 2018]) | détails exposés dans ce chapitre. |
| modélisation du segment | *TDNN* standard [Snyder *et al.*, 2018] ou réseau *ResNet-1D* (tableau 5.2) | entraîné par descente de gradient stochastique pour l'identification de quatorze langues avec l'entropie croisée, les augmentations de données et la stratégie *stochastic weight averaging* [Izmailov *et al.*, 2018] |
| classifieur final | non utilisé | utilisation directe des probabilités *a posteriori* produites par le réseau |
| calibration | régression logistique | modèles différents appris pour chaque durée de test |

TABLE 6.6 – Performance de reconnaissance de la langue sur les trois ensembles de test du corpus *NIST LRE 2007*, correspondant à différentes durées.

| *features* | Test - 3s (%) | | | Test - 10s (%) | | | Test - 30s (%) | | |
|---|---|---|---|---|---|---|---|---|---|
| | EER | minDCF | $C_{avg}$ | EER | minDCF | $C_{avg}$ | EER | minDCF | $C_{avg}$ |
| spectrogramme *Mel* | 15,25 | 11,50 | 13,76 | 10,03 | 6,98 | 9,02 | 10,06 | 6,29 | 9,03 |
| *BNF* de *BUT/Phonexia* | **7,93** | 7,31 | **8,12** | **2,43** | **1,80** | **2,40** | 1,49 | 0,85 | 1,37 |
| *BNF* du modèle *Conformer* | 8,01 | **6,81** | 8,42 | 3,08 | 1,84 | 2,53 | **0,91** | **0,38** | **0,77** |

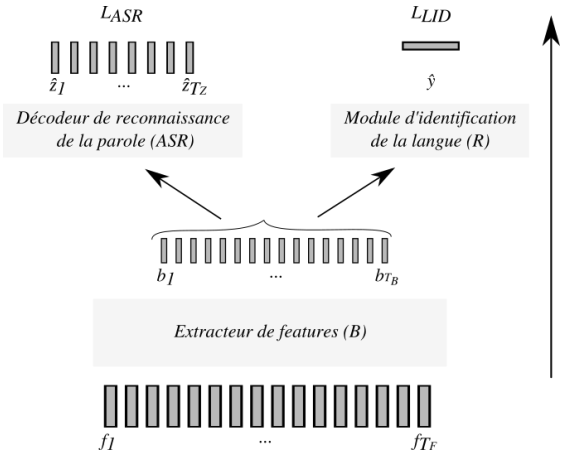



FIGURE 6.2 – Schéma du modèle conjoint d'identification de la langue et de reconnaissance de la parole.

## 6.3 Étude de la stratégie d'entraînement

La sélection de l'extracteur de *bottleneck features* est problématique. En effet ce module est entraîné avec une tâche de reconnaissance de la parole puis utilisé et évalué pour une tâche de reconnaissance de la langue. Il apparaît donc souhaitable d'adopter un point de vue intégré des entraînements du module *bottleneck features* et du réseau d'identification de la langue, dans le but de sélectionner les paramètres de l'extracteur de *bottleneck features* en fonction d'une performance d'identification de la langue.

### 6.3.1 Modèle conjoint d'identification de la langue et de reconnaissance de la parole

Dans ce but, nous introduisons maintenant un modèle conjoint d'identification de la langue et de la reconnaissance de la parole. Ce point de vue englobe à la fois les systèmes classiques de *bottleneck features* et les systèmes basés sur un modèle de reconnaissance de la parole *de bout en bout*.

Un modèle d'identification de la langue opère sur une séquence de *features* $f$ de longueur variable $T_F$ représentant le signal à l'échelle d'une trame. La tâche consiste à prédire une étiquette de reconnaissance de la langue $y$ parmi un ensemble fixé $\mathcal{Y}$. Pendant

l'entraînement du système, une information auxiliaire de reconnaissance de la parole peut être fournie sous la forme d'une séquence $Z$ de mots, phones ou caractères.

Une architecture conjointe de reconnaissance de la langue et de la parole est représentée en figure 6.2. Elle est composée de trois modules :

— un extracteur de *bottleneck features* ($B$) qui prend en entrée la séquence $f$ et produit une séquence $b$ d'*embeddings*
— un décodeur de reconnaissance automatique de la parole (*automatic speech recognition* - $ASR$) qui prend en entrée la séquence $b$ et prédit une séquence de scores de reconnaissance de la parole $\hat{Z}$
— un module d'identification de la langue ($R$) qui prend en entrée la séquence $b$ et produit un vecteur de scores d'identification de la langue $\hat{y}$

Les paramètres des deux modules peuvent être entraînés avec deux fonctions de coût différentes :

— une fonction de coût d'identification de la langue $L_{LID}(\hat{y},y)$
— une fonction de coût de reconnaissance de la parole $L_{ASR}(\hat{Z},Z)$

### 6.3.2 Choix d'architecture

Dans ce cadre, le remplacement des *bottleneck features* classiques par un modèle de reconnaissance de la parole *de bout en bout* revient à une modification de trois éléments : la séquence d'étiquettes $Z$ pour l'entraînement du modèle de reconnaissance de la parole, la fonction de coût $L_{ASR}$ utilisée pour l'entraînement de ce module et les architectures de l'extracteur de *features* $B$ et du décodeur de reconnaissance de la parole $ASR$.

### 6.3.3 Définition des stratégies

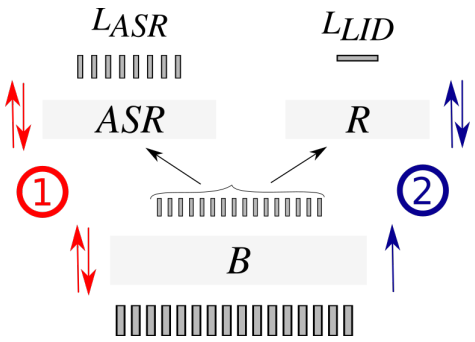
Une fois que nous avons rassemblé l'extracteur de *bottleneck features* et le module d'identification de la langue dans le même modèle, nous pouvons définir plusieurs stratégies d'entraînement.

#### Apprentissage en deux étapes (*2-étapes*)

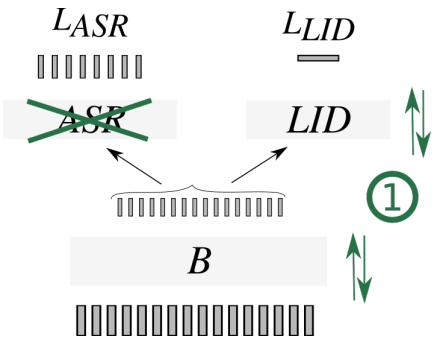
La stratégie d'entraînement classique pour les systèmes basés sur des *bottleneck features* comporte deux étapes. D'abord l'extracteur de *bottleneck features* et le module de reconnaissance de la parole sont entraînés avec la fonction de coût $L_{ASR}$. Ensuite les paramètres de l'extracteur de *bottleneck features* sont gelés et les paramètres du module d'identification de la langue sont appris en minimisant la fonction de coût $L_{LID}$. Elle est représentée en figure 6.3a.

#### Identification de la langue de bout en bout (*E2E LID*)

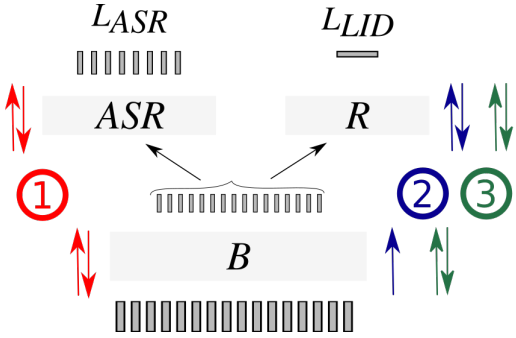
Une autre stratégie est l'entraînement simultané de l'extracteur de *bottlneck features* et du réseau d'identification de la langue. Dans ce cas l'entraînement de l'extracteur de *bottleneck features* se fait en minimisant la fonction de coût d'identification de la



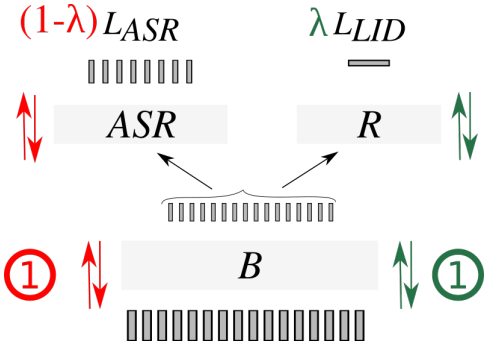

(a) Apprentissage en deux étapes (*2-étapes*)




(b) Identification de la langue de bout en bout (*E2E LID*)




(c) Apprentissage en deux étapes suivi d'un *finetuning* d'identification de la langue de bout en bout (*3-étapes*)




(d) Apprentissage multi-tâches

FIGURE 6.3 – Illustration des stratégies d'apprentissage pour l'entraînement de l'extracteur de *bottleneck features* $B$, le module d'identification de la langue $R$ et le décodeur de reconnaissance de la parole $ASR$. Les flèches indiquent les propagations avant et rétro-propagation et les numéros l'ordre des phases d'apprentissage.

langue. Cette stratégie ne comporte qu'une seule étape. Elle n'utilise pas les étiquettes de reconnaissance de la parole. La différence avec un modèle qui n'utilise pas de *bottleneck features* est l'ajout des couches de l'extracteur de *bottleneck features* dans le modèle d'identification de la langue. Elle est représentée en figure 6.3b.

#### Apprentissage en deux étapes suivi d'un *finetuning* d'identification de la langue de bout en bout (*3-étapes*)

Cette stratégie consiste en l'enchaînement des deux stratégies précédentes. Elle comporte donc trois étapes. D'abord l'extracteur de *bottleneck features* est entraîné avec la tâche de reconnaissance de la parole. Ensuite ses paramètres sont gelés et le module d'identification de la langue est entraîné avec la fonction de coût associée. Finalement, les paramètres de l'extracteur de *bottleneck features* sont dégelés et il est entraîné conjointement avec le module d'identification de la langue pour minimiser la fonction de coût d'identification de la langue. La stratégie est représentée en figure 6.3c.

Cette stratégie permet d'utiliser la tâche de reconnaissance de la parole pour initialiser l'extracteur de *bottleneck features* puis d'en raffiner les paramètres en fonction de la tâche d'identification de la langue. Elle suppose donc que les paramètres du modèle doivent être spécifiés pour la tâche mais que des représentations pertinentes peuvent être plus facilement découvertes par une tâche de reconnaissance de la parole qu'une tâche d'identification de la langue.

#### Apprentissage multi-tâches

L'apprentissage multi-tâche consiste à entraîner les trois modules en même temps en fonction d'un objectif prenant en compte les deux fonctions de coût [Punjabi *et al.*, 2021]. Cette stratégie est représentée en figure 6.3d. Plus précisément le module d'identification de la langue est entraîné pour minimiser la fonction de coût d'identification de la langue et le décodeur de reconnaissance de la parole à minimiser la fonction de coût de reconnaissance de la parole. L'extracteur de *bottleneck features* est entraîné avec la fonction de coût combinée (avec un poids $\lambda \in [0,1]$) :

$$L = \lambda L_{LID} + (1 - \lambda) L_{ASR} \tag{6.1}$$

Cette stratégie ne comporte qu'une seule étape d'entraînement. En revanche elle demande de sélectionner l'hyper-paramètre $\lambda$, éventuellement en le faisant varier au cours de l'entraînement.

### 6.3.4 Comparaison des stratégies

L'utilisation de *bottleneck features* peut bénéficier au système de reconnaissance de la langue de deux manières. D'abord, le réseau extracteur de *bottleneck features* ajoute des paramètres pouvant être entraînés au système et augmente donc sa capacité. D'autre part, la supervision avec la tâche de reconnaissance de la parole apporte une information complémentaire à la supervision par l'identification de la langue, ce qui peut améliorer la

qualité des représentations. En entraînant des systèmes avec différentes stratégies, nous voulons mettre en lumière les apports respectifs de ces deux effets.

### Système de reconnaissance de la langue utilisés

Nous réalisons des expériences avec le système de reconnaissance de la langue décrit dans le tableau 6.5. Deux modèles d'identification de la langue sont utilisés : le *TDNN* classique et une architecture *ResNet-1D* basée sur des convolutions à une dimension, décrite dans le tableau 5.2.

### Modèles d'extracteurs de *bottleneck features*

Nous réalisons des expériences avec deux modèles d'extracteurs de *bottleneck features*. Le premier modèle est un modèle classique entraîné pour une tâche de reconnaissance de phones à l'échelle d'une trame. Nous utilisons l'architecture de *stacked bottleneck features* basée sur l'enchaînement de deux perceptrons multi-couches proposée par [Fér *et al.*, 2017], nous le notons *DNN*. Pour les stratégies en deux et trois étapes, nous utilisons le modèle pré-entraîné de *BUT/Phonexia* sur dix-sept langues du corpus *Babel*. Le second modèle d'extracteur de *bottleneck features* est le modèle de reconnaissance de la parole *Conformer* associé à la méthode d'entraînement décrite dans ce chapitre. Les deux modèles produisent des *bottleneck features* de dimension 80.

### Corpus utilisés

Les modèles d'identification de la langue et les systèmes de reconnaissance de la langue sont entraînés et évalués pour le corpus *NIST LRE 2007* et la tâche de reconnaissance de quatorze langues. Le corpus *Babel* est utilisé pour la tâche de reconnaissance de la parole : dix-sept langues pour le modèle de *BUT/Phonexia* et quatorze langues pour le modèle de reconnaissance de la parole *Conformer*.

### Détails d'entraînement

Pour l'entraînement des extracteurs de *bottleneck features* et des modèles d'identification de la langue, nous utilisons des bancs de filtres en échelle *Mel* et la régularisation *specAugment*, sans autre augmentation de données. Nous sélectionnons le modèle de reconnaissance de la parole en fonction de la meilleure performance de validation tandis que la méthode *stochastic weight averaging* est appliquée au modèle d'identification de la langue. Pour chacune des architectures d'extracteur de *bottleneck features* et d'identification de la langue, nous entraînons un système avec les trois stratégies suivantes : identification de la langue de bout en bout, apprentissage en deux étapes et apprentissage en trois étapes. Comme point de référence, nous entraînons également chaque modèle d'identification de la langue directement sur les bancs de filtres en échelle *Mel*, sans utiliser de modèle d'extraction de *bottleneck features*. Les modèles d'identification de la langue sont entraînés avec des segments de trois secondes. Pour traiter les segments de durées nominales dix et trente secondes, le modèle d'identification de la langue est

raffiné pendant deux époques en minimisant une fonction de coût d'identification de la langue pour des segments de durée correspondante, et ce pour chaque système.

### Résultats

Chaque système est évalué pour les trois durées nominales de test du corpus *NIST LRE 2007*. Les résultats sont rapportés dans le tableau 6.7.

TABLE 6.7 – Performance de reconnaissance de la langue pour les trois ensembles de test du corpus *NIST LRE 2007* pour trois stratégies d'entraînement, deux extracteurs de *bottleneck features* et deux architectures d'identification de la langue.

| $R$ | $B$ | stratégie | Test - 3s (%) | | | Test - 10s (%) | | | Test - 30s (%) | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | EER | minDCF | $C_{avg}$ | EER | minDCF | $C_{avg}$ | EER | minDCF | $C_{avg}$ |
| TDNN | aucun | | 15,25 | 11,50 | 13,76 | 10,03 | 6,98 | 9,02 | 10,06 | 6,29 | 9,03 |
| | DNN | E2E LID | 11,43 | 10,03 | 11,34 | 5,58 | 4,85 | 5,93 | 3,03 | 2,28 | 2,92 |
| | | 2-étapes | 7,93 | 7,31 | 8,12 | **2,43** | 1,80 | 2,40 | 1,49 | 0,85 | **1,37** |
| | | 3-étapes | **7,21** | **6,53** | **7,38** | 2,56 | **1,72** | **2,28** | **1,30** | **0,79** | 1,61 |
| | Conformer | E2E LID | 7,30 | 6,05 | 7,21 | 2,73 | 2,28 | 3,17 | 1,53 | 0,80 | 1,27 |
| | | 2-étapes | 8,01 | 6,81 | 8,42 | 3,08 | 1,84 | 2,53 | **0,91** | **0,38** | **0,77** |
| | | 3-étapes | **6,00** | **5,48** | **5,91** | **2,38** | **1,63** | **2,28** | 0,98 | 0,52 | 1,05 |
| ResNet-1D | aucun | | 13,37 | 11,58 | 13,67 | 7,34 | 4,79 | 6,82 | 7,99 | 4,49 | 5,27 |
| | DNN | E2E LID | 11,92 | 10,49 | 11,55 | 3,73 | 3,11 | 3,69 | 2,13 | 0,90 | 1,79 |
| | | 2-étapes | 7,47 | 6,32 | 7,21 | **1,69** | **1,17** | **1,69** | **0,88** | **0,35** | **0,62** |
| | | 3-étapes | **6,24** | **5,56** | **6,25** | 1,97 | 1,51 | 2,16 | 1,44 | 0,59 | 1,55 |
| | Conformer | E2E LID | 6,51 | 5,67 | 6,56 | 1,63 | **1,02** | **1,50** | **0,70** | 0,33 | 0,59 |
| | | 2-étapes | 9,13 | 7,71 | 8,46 | 2,03 | 1,41 | 1,95 | 1,16 | 0,57 | 1,18 |
| | | 3-étapes | **5,76** | **5,12** | **5,93** | **1,48** | 1,11 | 1,58 | **0,70** | **0,21** | **0,53** |

Les résultats sur la comparaison des architectures d'extracteur de *bottleneck features* et des stratégies d'entraînement sont cohérents pour les modèles *TDNN* et *ResNet-1D*, avec de meilleures performances de reconnaissance de la langue pour l'architecture *ResNet-1D*. Nous tirons nos conclusions de l'analyse pour l'ensemble de test de durée trois secondes. Les performances pour les durées plus longues suivent les mêmes tendances globales avec quelques exceptions. Nous ne concluons pas sur ces résultats étant donné le fait que les différentes stratégies d'apprentissage ont été mises en œuvre pour des segments de trois secondes et que les performances pour des durées plus longues sont obtenues par raffinement des poids du réseau d'identification de la langue. Elles donnent donc une indication de la performance du système pour des durées plus longues, sans refléter forcément la performance de la stratégie d'entraînement appliquée à la durée correspondante.

Nous observons d'abord que, quelle que soit la stratégie d'entraînement, l'utilisation d'un extracteur de *bottleneck features* permet d'améliorer la performance par rapport à l'utilisation directe de *features* acoustiques. Pour la stratégie *E2E LID*, l'ensemble des paramètres du modèle ont été entraînés avec la tâche d'identification de la langue. Le gain en performance s'explique donc uniquement par l'ajout de couches au modèle d'identi-

fication de la langue. Le gain est beaucoup plus important pour le modèle *Conformer* ($C_{avg} \times 100$ de 6,56 au lieu de 13,67) que pour l'architecture *DNN* ($C_{avg} \times 100$ de 11,55).

D'autre part, pour l'architecture *DNN*, une amélioration très significative de performance est octroyée par la stratégie *2-étapes* (7,21) par rapport à la stratégie *E2E LID* (11,55). Cela n'est pas le cas pour le modèle *Conformer* (8,46 au lieu de 6,56).

Enfin, tant pour les modèles *DNN* que *Conformer*, la stratégie *3-étapes* atteint la meilleure performance (6,25 et 5,93), avec des taux d'erreur inférieurs aux stratégies *E2E LID* et *2-étapes*. La comparaison des quatre combinaisons d'architecture pour cette stratégie permet d'établir la supériorité du modèle *ResNet-1D* sur le *TDNN*, ainsi que celle du modèle *Conformer* sur les *DNN*.

Notons également que notre meilleure recette d'entraînement (*ResNet-1D* et *Conformer* avec la stratégie *3-étapes*) atteint une performance à l'état de l'art en termes de $C_{avg}$ pour les trois ensembles de test (5,93, 1,58 et 0,53), comparée à [Ling *et al.*, 2020] pour des segments de trois et dix secondes et [Cai *et al.*, 2020b, Cai *et al.*, 2018a] pour des segments de trente secondes.

**Conclusion sur les stratégies d'entraînement**

Un comportement différent apparaît donc entre le modèle classique de *bottleneck features* et le modèle de reconnaissance de la parole *de bout en bout*. Le modèle classique pose des problèmes d'optimisation avec la stratégie *E2E LID*. La majeure partie du gain de performance vient de la supervision de ce module par la tâche de reconnaissance de phones. À l'inverse, le modèle de reconnaissance de la parole *Conformer* peut être entraîné de façon très efficace avec la fonction de coût d'identification de la langue. Ce résultat n'est pas surprenant, le modèle *Conformer* étant une architecture *de séquence à séquence* adaptée à une unique étiquette pour l'ensemble d'un segment, contrairement à l'architecture *DNN* qui fonctionne à l'échelle d'une trame. Par conséquent, pour le modèle de reconnaissance de la parole *de bout en bout* la stratégie *2-étapes* est moins performante que la stratégie *E2E LID* bien qu'elle autorise un gain significatif par rapport à l'usage direct de bancs de filtres.

Nous en déduisons plusieurs enseignements. Pour tous les modèles de *bottleneck features*, l'ajout de paramètres au systèmes et la supervision supplémentaire contribuent au gain de performance. L'importance de la supervision est plus forte pour le modèle classique tandis que l'apport de l'architecture domine pour le modèle de reconnaissance de la parole *de bout en bout*. Pour les deux types de modèles, la meilleure stratégie est le pré-entraînement des *bottleneck features* avec une tâche de reconnaissance de la parole puis un raffinement avec la tâche d'identification de la langue. Nous en concluons que la tâche de reconnaissance de la parole est utile au début de l'entraînement pour la découverte de motifs inaccessibles à la tâche d'identification de la langue.

### 6.3.5 Apprentissage multi-tâches

Nous évaluons la stratégie multi-tâches pour notre meilleure combinaison d'architectures : les modèles *ResNet-1D* et *Conformer*. Cette stratégie a l'avantage d'un appren-

tissage simplifié puisqu'elle ne comporte qu'une seule étape. Elle nécessite cependant de choisir la valeur de l'hyper-paramètre $\lambda \in [0,1]$.

Nous entraînons trois systèmes avec différentes valeurs de $\lambda$. Nous utilisons d'abord des valeurs fixes de $\lambda$. $\lambda = 0{,}5$ est utilisé naïvement afin d'équilibrer les deux fonctions de coût, tandis que $\lambda = 0$ permet de nous comparer à la stratégie *2-étapes*.

Forts du constat que la tâche de reconnaissance de la parole est utile au début de l'apprentissage, nous entraînons un système en augmentant linéairement la valeur de $\lambda$ au cours de l'apprentissage entre 0 et 1.

### Détails d'implémentation

Tous les entraînements sont réalisés sur cent époques, comme pour l'entraînement de l'extracteur de *bottleneck features* pour la stratégie *2-étapes*. Pour l'augmentation linéaire du paramètre $\lambda$, la valeur est augmentée de $\frac{1}{99}$ à la fin de chaque époque.

Le modèle final est sélectionné avec la stratégie *stochastic weight averaging*. Notons que la sélection d'un modèle en fonction de la meilleure fonction de coût sur l'ensemble de validation serait problématique lorsque le paramètre $\lambda$ varie au cours de l'apprentissage puisque les fonctions de coût calculées avec différentes valeurs de $\lambda$ ne sont pas comparables. La sélection du modèle en fonction du coût d'identification de la langue uniquement reste possible mais est en contradiction avec le principe de l'apprentissage multi-tâches.

### Résultats

Les performances des stratégies d'entraînement multi-tâches sont présentées dans le tableau 6.8.

TABLE 6.8 – Performance de reconnaissance de la langue pour les trois ensembles de test du corpus *NIST LRE 2007* pour un entraînement multi-tâches. Les modèles *ResNet-1D* et *Conformer* sont utilisés.

| stratégie | $\lambda$ | Test - 3s (%) | | | Test - 10s (%) | | | Test - 30s (%) | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | EER | minDCF | $C_{avg}$ | EER | minDCF | $C_{avg}$ | EER | minDCF | $C_{avg}$ |
| E2E LID | | 6,51 | 5,67 | 6,56 | 1,63 | **1,02** | **1,50** | **0,70** | 0,33 | 0,59 |
| 2-étapes | | 9,13 | 7,71 | 8,46 | 2,03 | 1,41 | 1,95 | 1,16 | 0,57 | 1,18 |
| 3-étapes | | **5,76** | **5,12** | **5,93** | **1,48** | 1,11 | 1,58 | **0,70** | **0,21** | **0,53** |
| multi-tâches | 0 | 8,09 | 6,89 | 8,02 | 2,05 | 1,07 | 1,77 | 1,02 | 0,31 | 0,89 |
| | 0,5 | 5,99 | **4,81** | 5,91 | 2,13 | 1,16 | 2,03 | 1,26 | 0,67 | 1,31 |
| | $0 \rightarrow 1$ | **5,58** | 4,89 | **5,78** | 1,75 | 1,12 | 1,79 | 1,02 | 0,50 | 0,71 |

Notons d'abord que la stratégie multi-tâches avec $\lambda = 0$ aboutit à une meilleure performance ($C_{avg} \times 100$ de 8,02) que la stratégie *2-étapes* (8,46) alors que l'extracteur de *bottleneck features* suit la même dynamique pour ces deux stratégies. Cela signifie que le module d'identification de la langue atteint une meilleure configuration s'il est adapté en permanence à l'extracteur de *bottleneck features* au cours de l'entraînement de celui-ci

que s'il est entièrement entraîné une fois que les paramètres de l'extracteur de *bottleneck features* sont fixés. Ce résultat à lui seul plaide pour l'intégration des modules du système de reconnaissance de la langue pour les entraîner conjointement.

D'autre part, la stratégie multi-tâches avec la valeur naïve $\lambda = 0{,}5$ atteint une performance équivalente ($C_{avg} \times 100 = 5{,}91$) à la stratégie *3-étapes* (5,93) pour des segments de trois secondes, mais avec une recette d'entraînement simplifiée ne comportant qu'une étape, ce qui la rend particulièrement attractive.

Enfin, l'augmentation de $\lambda$ au cours de l'apprentissage ($0 \rightarrow 1$) permet d'améliorer légèrement la performance en terme de $C_{avg} \times 100$ (5,78). Cela valide l'hypothèse selon laquelle la supervision par la tâche de reconnaissance de la parole est principalement utile au début de l'entraînement pour découvrir des motifs tandis que le modèle gagne à être raffiné en fonction de la tâche visée.

Il serait intéressant de développer des stratégies plus sophistiquées d'évolution de $\lambda$ au cours de l'apprentissage. On pourrait penser à une évolution exponentielle du poids [Ganin *et al.*, 2016] ou à une valeur dépendant de la progression de l'apprentissage [Chen *et al.*, 2018].

## 6.4 Conclusion

Les systèmes de reconnaissance de la langue à l'état de l'art utilisent des *bottleneck features*, ce qui leur permet de bénéficier d'une supervision par une tâche de reconnaissance de la parole. Deux dynamiques sont à l'œuvre dans le domaine des *bottleneck features*. D'abord, les architectures classiques de reconnaissance de phones à l'échelle d'une trame sont remplacées par des modèles de reconnaissance de la parole *de bout en bout* qui, tout en atteignant une meilleure performance, présentent une recette d'entraînement simplifiée. D'autre part, des travaux majoritairement tournés vers les architectures classiques visent à intégrer l'entraînement de l'extracteur de *bottleneck features* et du module d'identification de la langue.

Dans un premier temps, nous réalisons pour la première fois à notre connaissance l'entraînement d'un modèle de reconnaissance de la parole *de bout en bout* multilingue dans le but d'extraire des *bottleneck features*, et ce avec une architecture moderne, *Conformer*, introduite en 2020. Nous montrons que cette approche atteint la même performance que des *bottleneck features* classiques, alors même que la recette d'apprentissage est grandement simplifiée.

Dans le but de réaliser un entraînement conjoint des deux modules, nous étudions la stratégie d'entraînement, à la fois pour une architecture classique et pour un modèle de reconnaissance de la parole *de bout en bout*. Cette analyse révèle que la meilleure stratégie est la même pour les deux types d'architectures. Elle comporte trois étapes : entraînement de l'extracteur de *bottleneck features* avec la tâche de reconnaissance de la parole, puis entraînement du module d'identification de la langue avec la tâche d'identification de la langue et enfin raffinement conjoint des deux modules avec la tâche d'identification de la langue. Pour les deux types d'architectures, l'amélioration de la performance est rendue possible, à la fois par l'ajout de couches supplémentaires au modèle de reconnaissance de

la langue et par la supervision par la tâche de reconnaissance de la parole. Cependant la comparaison des stratégies d'entraînement pour les différents modèles montre que c'est la supervision par la tâche de reconnaissance de la parole qui apporte le gain le plus important pour les *bottleneck features* classiques tandis que c'est l'amélioration de l'architecture du système pour le modèle de *bottleneck features* de bout en bout. Avec cette stratégie d'apprentissage en trois étapes, optimale pour tous les modèles, le modèle de reconnaissance de la parole *de bout en bout* est nettement supérieur à l'architecture classique.

Une performance identique peut être atteinte avec un entraînement multi-tâches en une seule étape, et ce avec une valeur fixe des poids respectifs des deux fonctions de coût. Nous montrons que la performance de cette stratégie peut être améliorée en imitant la stratégie en trois étapes, c'est-à-dire en augmentant le poids de la tâche d'identification de la langue au cours de l'apprentissage. Nous pensons que des gains supplémentaires peuvent être espérés par le développement de stratégies plus sophistiquées d'évolution de ce poids.

Enfin, nous avons montré que l'application de méthodes de régularisation lors de l'entraînement du modèle de reconnaissance de la parole *de bout en bout* permet d'augmenter la performance du système de reconnaissance de la langue. Un travail restant à mener est l'application des méthodes de régularisation développées dans le chapitre 4 à ce module du système. Rappelons que nous avons montré que l'adaptation au canal de transmission du module d'identification de la langue était plus efficace que celle du classifieur final. Il est donc légitime de vouloir remonter encore dans le système en adaptant au canal l'extracteur de *bottleneck features*.

# 7

# Conclusion

Cette thèse s'intéresse à la robustesse au canal de transmission des systèmes de reconnaissance de la langue. C'est un problème largement étudié dans la littérature et non résolu. De nombreuses méthodes ont été proposées, dont beaucoup dépendent des architectures des systèmes utilisés pour réaliser la tâche. Depuis les années 2017-2018, le type de système dominant repose sur un réseau de neurones profond réalisant la tâche d'identification de la langue pour l'ensemble du segment audio, soit pour produire directement des scores, soit pour extraire une représentation du signal appelée *embedding*. Nous explorons des approches d'amélioration de la robustesse au canal de transmission adaptées aux modèles à base de réseaux de neurones.

Nous adoptons un point de vue sur le problème de la robustesse centré sur la qualité des représentations utilisées par le système, principalement dans l'espace des *embeddings*. Nous considérons que la limitation de l'impact du canal de transmission sur les performances des systèmes peut être obtenue par une augmentation de l'invariance de ces représentations entre les canaux de transmission, compris comme des domaines. Dès lors, deux directions de recherche émergent : le développement de méthodes d'augmentation de l'invariance des représentations et l'analyse de ces représentations dans le but de caractériser la robustesse des systèmes.

Nous avons proposé et étudié une méthode d'augmentation de l'invariance des représentations : la régularisation de la fonction de coût du réseau d'identification de la langue avec une fonction qui vise à minimiser l'écart entre les représentations des différents domaines. Deux types de fonctions de régularisation ont été explorées dans cette thèse : des mesures de divergence entre les domaines et des fonctions de *metric learning*. Signalons que la régularisation du réseau de neurones principal d'un système de reconnaissance de la langue ou de reconnaissance du locuteur dans le but de réduire l'effet d'un changement de domaine a fait l'objet de nombreux travaux concomitants aux nôtres dans les années 2018 à 2021, avec différentes propositions pour les fonctions de régularisation et les types de variabilités étudiés. Pour notre part, nous avons montré l'intérêt de cette méthode

dans trois scénarios d'apprentissage correspondant à des situations d'intérêt pratique où la diversité des canaux de transmission a un effet négatif sur la performance.

L'adaptation de domaine non supervisée est un scénario d'apprentissage où le canal de transmission visé n'est accessible lors de l'entraînement du système qu'à travers des données non annotées. Dans le cadre de ce scénario, nous avons comparé plusieurs fonctions de régularisation et évalué le gain apporté par une régularisation sur différentes couches du réseau de neurones d'identification de la langue. La méthode donnant les meilleurs résultats est basée sur une fonction de régularisation appelée *maximum mean discrepancy*, appliquée sur la couche de sortie du réseau. Ses hyper-paramètres sont robustes aux canaux de transmission sur lesquels la méthode est mise en œuvre, et elle peut être appliquée conjointement sur plusieurs canaux cibles. Avec cette régularisation du réseau de neurones central, nous obtenons une performance d'adaptation supérieure aux méthodes existantes d'adaptation d'autres modules du système. Les expériences menées montrent même que cette méthode d'adaptation de domaine non supervisée atteint de meilleures performances qu'un système entraîné de façon supervisée sur le domaine visé. Cela signifie que la régularisation permet d'obtenir des représentations de meilleure qualité en bénéficiant des différents canaux de transmission présents dans l'ensemble d'entraînement.

L'apprentissage multi-domaines est un scénario où cette dernière propriété doit être mise en œuvre. Plusieurs canaux de transmission sont représentés dans l'ensemble d'entraînement et il s'agit de produire un système atteignant une bonne performance sur l'ensemble de ceux-ci. Nous avons montré que les deux familles de fonctions de régularisation basées sur une mesure de divergence ou sur le *metric learning* peuvent être appliquées avec succès pour ce scénario. Ces deux types de fonctions ont des effets différents sur l'espace des représentations. Les mesures de divergence réduisent la variabilité des représentations due à un facteur de variabilité identifié. Elles nécessitent donc l'emploi d'annotations en canal de transmission. En revanche elles peuvent être appliquées sans annotation en langue. À l'inverse, les fonctions de *metric learning* opèrent uniquement sur des données annotées en langue, mais n'ont pas besoin d'annotations en canal. Le *metric learning* réduit de façon agnostique l'ensemble des variabilités nuisibles pour la tâche de classification rencontrées dans le corpus d'entraînement, y compris le canal de transmission.

Enfin nous avons mis en œuvre les fonctions de régularisation pour la généralisation à un canal de transmission inconnu. Dans ce scénario, les fonctions de régularisation, en imposant une invariance des représentations par rapport aux canaux rencontrés dans l'ensemble d'entraînement, sont supposées leur octroyer une robustesse à de nouveaux canaux inconnus attendus lors de l'inférence. Nous avons exploré ce scénario à l'occasion de notre participation à la compétition *Oriental Language Recognition 2020*, pour laquelle nous avons obtenu la première place pour deux des trois tâches proposées. Dans ce cadre, les méthodes de régularisation du réseau de neurones d'identification de la langue nous ont permis de construire des systèmes robustes aux conditions inconnues mais peu performants. Ces systèmes permettent cependant une amélioration très nette de la performance sur les canaux inconnus lorsqu'ils sont utilisés en combinaison avec des systèmes

plus performants mais moins robustes. Il est remarquable que les méthodes de régularisation développées pour augmenter la robustesse au canal de transmission (tâche 1 de la compétition) ont également permis d'augmenter la robustesse à des conditions bruitées (tâche 3).

Au cours de ce travail, à la fois dans le but de mettre au point les méthodes de régularisation et pour rendre compte des propriétés des représentations obtenues, nous avons développé des méthodes d'analyse de l'espace des représentations. Ces méthodes visent à mesurer la variabilité des représentations due au canal de transmission et à la mettre en rapport avec la discriminabilité des représentations correspondant à différentes langues. Des représentations invariantes sont des vecteurs pour lesquels la variabilité due au canal est négligeable par rapport à la variabilité correspondant à la langue. Nous avons proposé deux outils de mesure de ces variabilités : le rapport entre les covariances inter-classes et intra-classes et la mesure de divergences.

Ces outils nous octroient une compréhension plus fine des systèmes. Ils nous ont permis de comprendre la différence de nature entre les fonctions de régularisation à base de mesures de divergence et celles à base de *metric learning*, ainsi que de réaliser que la régularisation du réseau produit des systèmes intrinsèquement différents d'un apprentissage multi-domaines. Un apprentissage multi-domaines préserve des représentations différenciées pour les différents canaux sur des couches profondes du réseau de neurones d'identification de la langue, tandis que la régularisation, même appliquée sur la couche de sortie, impose une invariance au canal sur les couches intermédiaires du réseau. Ce comportement explique que la régularisation du réseau dans un scénario d'adaptation de domaine non supervisée permet parfois d'améliorer la performance par rapport à un apprentissage supervisé sur le domaine cible.

À l'occasion de notre participation au *workshop SCALE 2020*, nous avons appliqué ce type d'analyse à la tâche de reconnaissance de la parole. Nous avons ainsi proposé de donner une estimation non supervisée de la performance de réseaux de reconnaissance de la parole de bout en bout sur de nouveaux domaines en fonction de mesures de divergence dans un espace d'*embeddings* (travail présenté en annexe E). Ces résultats illustrent le fait que les outils d'analyse des variabilités dans un espace de représentations, proposés dans cette thèse pour la reconnaissance de la langue, sont génériques et applicables à d'autres systèmes reposant sur des réseaux de neurones profonds.

La dernière direction de notre travail porte sur l'amélioration d'un autre module du système, l'extracteur de *bottleneck features*. Nous avons d'abord étendu les travaux existants sur l'utilisation d'un réseau de neurones de reconnaissance de la parole de bout en bout pour réaliser cette extraction, en proposant un modèle multilingue. Cette approche permet d'éviter la phase d'alignement forcé des trames acoustiques avec les transcriptions, ainsi que de se passer de dictionnaires de prononciation. Elle est donc plus facile à mettre en œuvre que les modèles classiques d'extraction de *bottleneck features*, tout en atteignant des performances comparables. De plus, la performance de reconnaissance de la langue permise par ces *bottleneck features* peut être améliorée en appliquant des

régularisations lors de l'entraînement du réseau de reconnaissance de la parole de bout en bout.

D'autre part, nous avons analysé la stratégie d'apprentissage d'un système constitué d'un extracteur de *bottleneck features* et d'un réseau d'identification de la langue, tant pour des *bottleneck features* classiques que pour un modèle de bout en bout. Pour les deux types de modèles, la meilleure stratégie d'apprentissage est constituée de trois étapes : entraînement de l'extracteur de *bottleneck features* avec une tâche de reconnaissance de la parole, entraînement du réseau d'identification de la langue, puis raffinement conjoint des deux modules en fonction d'un objectif d'identification de la langue. Nous avons ensuite montré que la même performance peut être atteinte en une seule phase d'apprentissage par un entraînement multi-tâches de l'ensemble du système. De plus, une stratégie d'évolution des poids relatifs des deux tâches de reconnaissance de la parole et d'identification de la langue au cours de l'entraînement peut encore améliorer la performance.

Les trois directions explorées au cours de cette thèse, régularisation du réseau d'identification de la langue, analyse de l'espace des représentations et amélioration de la qualité des *bottleneck features*, adoptent le même point de vue : l'augmentation de la performance du système par un travail sur la qualité de ses représentations. La compréhension des représentations produites par des réseaux de neurones profonds reste ouverte. Notre contribution dans cette direction se limite à l'évaluation de l'impact de facteurs de variabilité connus sur les représentations. Peut-on aller plus loin et identifier les principaux facteurs de variabilité qui affectent la performance par une analyse de l'espace des représentations ? Une autre question d'intérêt pratique, que nous avons commencé à explorer pour la tâche de reconnaissance de la parole, est la prédiction non supervisée de performance sur un nouveau domaine.

Deux dynamiques principales sont à l'œuvre dans le domaine de la reconnaissance de la langue : le remplacement de la plupart des modules du système par des réseaux de neurones profonds et l'intégration des modules du système dans la direction d'un unique modèle réalisant la tâche de bout en bout. En développant des outils qui augmentent la robustesse de tels modèles et en proposant l'intégration de l'extracteur de *bottleneck features* avec le réseau d'identification de la langue, nous participons pleinement à ces deux dynamiques. Nous espérons donc que les méthodes d'augmentation de la robustesse et d'analyse de la qualité des représentations, dont l'efficacité a été démontrée au cours de cette thèse, demeureront pertinentes à moyen terme pour les nouveaux systèmes de référence de reconnaissance de la langue qui ne manqueront pas d'émerger. Tant que les systèmes reposent sur un réseau de neurones entraîné pour une tâche d'identification de la langue, la méthode de régularisation de la fonction de coût reste applicable, avec peut-être un impact plus grand si la tâche est réalisée de bout en bout par un unique système.

La méthode de régularisation explorée au cours de cette thèse est complémentaire

d'autres approches d'augmentation de la robustesse au canal de transmission. Les augmentations de données peuvent être raffinées dans ce but et il est toujours envisageable de réaliser un traitement sur les enregistrements lors de l'inférence pour limiter l'effet du canal de transmission sur le signal. Émergent également pour la tâche de reconnaissance de la parole des méthodes de pré-entraînement non supervisé d'extracteurs de *features* [Baevski *et al.*, 2020]. Ces approches sont particulièrement intéressantes dans un scénario d'adaptation de domaine non supervisée car elles permettent d'utiliser des données non annotées lors de la phase de pré-apprentissage.

Enfin d'autres fonctions de régularisation peuvent être explorées dans le cadre de la méthode que nous avons étudiée. Parmi les mesures de l'écart entre deux domaines utilisées pour des scénarios d'adaptation de domaine dans la littérature, relevons l'apprentissage antagoniste et le transport optimal. D'autre part, nous nous sommes limités à des fonctions de coût imposant une proximité entre des représentations provenant de différents domaines. Un point de vue légèrement différent vise à limiter l'effet du canal par une normalisation basée sur une analyse de la variabilité due au canal au sein d'un enregistrement [Cai *et al.*, 2020a].

Enfin, nous avons montré que l'intégration de l'entraînement de l'extracteur de *bottleneck features* et du réseau d'identification de la langue apporte un gain de performance et proposé une première stratégie d'évolution des poids des deux tâches lors de leur entraînement conjoint. Il est probable qu'une stratégie plus sophistiquée améliore encore la performance de reconnaissance de langue. De plus, l'entraînement conjoint de ces deux modules autorise l'application simultanée des régularisations étudiées dans cette thèse aux deux modules. Nous avons montré que l'adaptation au canal de transmission du réseau d'identification de la langue était supérieure à l'adaptation du classifieur final. Il est donc envisageable que l'adaptation d'un module encore antérieur, l'extracteur de *bottleneck features*, améliore encore la robustesse.

# A

# Description des corpus

Les systèmes à base d'apprentissage automatique reposent sur des corpus, mobilisés pour leur entraînement et leur évaluation. Nous décrivons ici les corpus utilisés dans notre travail. Il s'agit de corpus standards de la communauté académique de reconnaissance de la langue.

## A.1 *NIST Language Recognition Evaluations*

Depuis 1996, les évaluations de reconnaissance de la langue organisées par le *National Institute of Standards and Technology* (*NIST*, États-Unis d'Amérique) ont dominé la recherche pour cette tâche. Lors de chaque évaluation, un corpus d'entraînement et un corpus de test sont définis.

### A.1.1 Description des corpus d'origine

Nous constituons des ensembles d'entraînement et de test à partir des corpus des campagnes *NIST Language Recongition Evaluation* (*LRE*), *Callfriend* et *NIST Speaker Recognition Evaluation 2008* (*NIST SRE 2008*). Nous ajoutons les corpus *Callfriend* et *NIST SRE 2008* aux corpus *NIST LRE* afin d'être conformes à la littérature récente.

Ces corpus sont décrits dans les tableaux A.1, A.2 et A.3. Ils contiennent des données enregistrées pour deux canaux de transmission : le téléphone et *BNBS* (*broadcast narrowband speech*). Le canal *BNBS* correspond à des interventions téléphoniques retransmises lors d'émissions télévisées ou de radio. Une grande partie des enregistrements *BNBS* de ces corpus provient de la chaîne de radio internationale *Voice of America*. Pour chacun des corpus *LRE*, les ensembles de test sont partagés en trois ensembles, correspondant aux durées nominales de parole : trois, dix et trente secondes.

### A.1.2 Constitution des corpus utilisés dans nos expériences

Nous entraînons des systèmes en visant deux corpus d'évaluation : l'évaluation *NIST LRE 2007* et l'évaluation *NIST LRE 2011*.

TABLE A.1 – Caractéristiques des corpus *NIST LRE*.

| Corpus | Langues | Canaux | Taille (durée et nombre de fichiers) |
|---|---|---|---|
| LRE2003 (LDC2006S31) | allemand, anglais, arabe, coréen, espagnol, farsi, français, hindi, japonais, mandarin, russe, tamoul, vietnamien | téléphone | 46 heures, 11830 |
| LRE2005 (LDC2008S05) | anglais (États-Unis et Inde), coréen, espagnol (accent mexicain), hindi, japonais, mandarin (continental et de Taïwan), tamoul | téléphone | 73 heures, 11106 |
| LRE2007-train (LDC2009S05) | arabe (Égypte), bengali ,cantonais, espagnol, mandarin, minnan, ourdou, russe, tamoul, thaï, wu | téléphone | 118 heures, 320 |
| LRE2007 (LDC2009S04) | allemand, anglais, arabe, arabe standard, bengali, cantonais, coréen, dari, espagnol, farsi, hindi, japonais, mandarin, russe, tamoul, thaï, vietnamien | téléphone | 66 heures, 7530 |
| LRE2009 (LDC2014S06) | amharique, anglais, bosniaque, cantonais, coréen, croate, dari, espagnol, farsi, français, géorgien, haïtien, hausa, hindi, mandarin, ourdou, pachto, portugais, russe, turc, ukrainien, vietnamien | téléphone et *BNBS* | 215 heures, 41793 |
| LRE2011 (LDC2018S06) | anglais (États-Unis et Inde), arabe (Iraq, Levant, Maghreb, standard), bengali, dari, espagnol, farsi, hindi, lao, mandarin, ourdou, pachto, pendjabi, polonais, russe, slovaque, tamoul, tchèque, thaï, turc, ukrainien | téléphone et *BNBS* | 204 heures, 30804 |

TABLE A.2 – Caractéristiques du corpus *Callfriend*.

| | |
|---|---|
| Canal de transmission | téléphone |
| Durée | plus de 25 heures par langue |
| Nombre de fichiers | 60 conversations par langue |
| Langues | allemand, anglais (sud des États-Unis et reste des États-Unis), arabe (Égypte), coréen, espagnol (Caraïbes et reste de l'Amérique Latine), farsi, français (Canada), hindi, japonais, mandarin (continental et Taïwan), tamoul, vietnamien |
| Références LDC | LDC96S49, LDC96S50, LDC96S51, LDC96S52, LDC96S53, LDC96S54, LDC96S55, LDC96S56, LDC96S57, LDC96S58, LDC96S59, LDC96S60, LDC2014S01, LDC2018S09, LDC2019S04, LDC2019S18, LDC2019S21, LDC2020S06, LDC2020S08 |

TABLE A.3 – Caractéristiques du corpus d'entraînement *NIST SRE 2008*.

| | |
|---|---|
| Canal de transmission | téléphone et microphone |
| Durée | 1590 heures |
| Langues | anglais, arabe (Égypte et Maroc), bengali, coréen, dari, espagnol, farsi, géorgien, hindi, italien, japonais, khmer, lao, mandarin, minnan, ourdou, pendjabii, russe, tagalog, thaï, tigrigna |
| Détails | la majorité du corpus est en anglais |
| Références LDC | LDC2011S05 et LDC2011S07 |

**Expériences visant le corpus *NIST LRE 2007***

Le corpus de test est celui de l'évaluation *NIST LRE 2007* [NIST, 2007]. Nous traitons la tâche de reconnaissance de la langue dans un ensemble fermé avec les quatorze langues cibles suivantes : allemand, anglais, arabe, bengali, chinois, coréen, espagnol, farsi, hindoustani (regroupement de l'hindi et de l'ourdou), japonais, russe, tamoul, thaï, vietnamien.

Nous utilisons comme corpus d'entraînement les corpus *NIST LRE 2003*, *2005*, *2007 train*, *2009* (seulement les enregistrements téléphoniques), *Callfriend*, et un sous-ensemble des données téléphoniques du corpus *NIST SRE 2008* (un maximum de soixante conversations par langue). Cet ensemble d'entraînement est le même que dans des travaux récents en reconnaissance de la langue [Cai *et al.*, 2020b, Ling *et al.*, 2020]. Seules les quatorze langues cibles sont utilisées pour entraîner le système.

**Expériences visant le corpus *NIST LRE 2011***

Le corpus de test est celui de l'évaluation *NIST LRE 2011* [NIST, 2013]. Nous traitons la tâche de reconnaissance de la langue parmi vingt-quatre langues cibles. Le corpus d'entraînement rassemble les corpus *NIST LRE 2003*, *2005*, *2007*, *2009* et le sous-ensemble dédié à l'entraînement du corpus *NIST LRE 2011*. Le corpus d'entraînement comprend cinquante et une langues.

La particularité de cette configuration expérimentale est la présence de deux canaux de transmission, téléphone et *BNBS*, dans les ensembles d'entraînement et de test. Si vingt des vingt-quatre langues cibles sont évaluées sur les deux canaux, la plupart des langues ne sont observées que sur un des deux canaux au cours de l'entraînement. La répartition des langues en fonction des canaux dans les ensembles de test et d'entraînement est précisée dans le tableau A.4.

Lors de l'analyse des systèmes dans le chapitre 5, nous regroupons les vingt langues présentes sur les deux canaux dans l'ensemble d'évaluation en fonction des canaux présents dans l'ensemble d'entraînement. Les trois groupes sont nommés $G_A$ (langues observées sur les deux canaux dans l'ensemble d'entraînement), $G_T$ (langues observées uniquement sur le canal téléphonique) et $G_B$ (langues observées uniquement sur le canal *BNBS*).

## A.2 Robust Automatic Transcription of Speech (*RATS*)

### A.2.1 Description des corpus d'origine

Nous utilisons des enregistrements provenant de deux corpus : *RATS SAD* (*speech activity detection*, détection d'activité vocale) décrit dans le tableau A.5 et *RATS KWS* (*keyword spotting*, détection de mots-clefs) décrit dans le tableau A.6. Le corpus *OpenSAD 15* est un sous-ensemble du corpus *RATS* que nous avons utilisé au début de notre thèse avant d'avoir acquis les corpus *RATS SAD* et *RATS KWS*. Il est décrit dans le tableau A.7.

TABLE A.4 – Distribution des langues du corpus *NIST LRE 2011* en fonction de leur présence sur les deux canaux dans les ensembles d'entraînement et de test. $G_A$, $G_T$ et $G_B$ sont trois groupes de langues utilisés pour l'analyse des systèmes dans le chapitre 5.

| Présence dans l'ensemble d'entraînement | Présence dans l'ensemble de test | | Absence dans l'ensemble de test |
|---|---|---|---|
| | sur les deux canaux | uniquement sur le canal d'entraînement | |
| sur les deux canaux | **$G_A$** : anglais (États-Unis), farsi, hindi, mandarin, russe, espagnol, ourdou | | arabe (Égypte), arabe (dialecte inconnu), cantonais, français, coréen, vietnamien |
| uniquement téléphone | **$G_T$** : anglais (Inde), bengali, lao, pendjabi, polonais, slovaque, tamoul, tchèque, thaï | arabe (Iraq), arabe (Levant), arabe (Maghreb) | allemand, indonésien, italien, japonais, minnan, tagalog, wu |
| uniquement *BNBS* | **$G_B$** : dari, pachto, turc, ukrainien | arabe (standard) | amharique, azéri, biélorusse, bosniaque, bulgare, croate, géorgien, haïtien, hausa, ouzbek, portugais, roumain, swahili, tibétain |

TABLE A.5 – Caractéristiques du corpus *RATS SAD*

| | |
|---|---|
| Canaux de transmission | téléphone et huit canaux radios : D et H (HF), E (VHF), A, B, C, F, G (UHF) |
| Durée | 3000 heures |
| Langues | anglais, arabe (syro-libanais du nord et du sud), farsi, ourdou, pachto |
| Référence LDC | LDC2015S02 |

TABLE A.6 – Caractéristiques du corpus *RATS KWS*

| | |
|---|---|
| Canaux de transmission | téléphone et huit canaux radios : D et H (HF), E (VHF), A, B, C, F, G (UHF) |
| Durée | 3100 heures |
| Langues | arabe (syro-libainais du nord et du sud), farsi |
| Référence LDC | LDC2017S10 |

Le corpus *RATS* est un corpus parallèle : les mêmes enregistrements collectés sur un canal téléphonique (appelée *src*) sont transmis à travers huit canaux radios : D et H (HF), E (VHF), A, B, C, F, G (UHF). Le corpus *OpenSAD 15* ne dispose pas des canaux A et C. Une segmentation en activité vocale est fournie avec le corpus. Il est également divisé en trois sous-ensembles correspondant à l'entraînement (*train*), la validation (*dev1*) et le test (*dev2*).

### A.2.2 Constitution des corpus utilisés dans nos expériences

Pour toutes nos expériences avec ces corpus nous utilisons des segments de trois secondes. Nous fusionnons les deux corpus *RATS SAD* et *RATS KWS* pour former un unique corpus contenant cinq langues (anglais, arabe, farsi, ourdou, pachto) que nous appelons *RATS*. Nous regroupons les deux dialectes d'arabe (syro-libanais du nord et du sud) en une seule classe appelée arabe pour nos expériences de reconnaissance de la langue. Deux sous-ensembles sont utilisés : *OpenSAD 15* comprenant sept canaux de transmission (src, B, D, E, F, G, H) et *RATS* qui comprend les neuf canaux. Comme la quantité de données de farsi est faible dans le corpus *OpenSAD 15*, nous n'utilisons

TABLE A.7 – Caractéristiques du corpus *OpenSAD 15*

| | |
|---|---|
| Canal de transmission | téléphone et six canaux radios : D et H (HF), E (VHF), B, F, G (UHF) |
| Durée | 1883 heures |
| Langues | anglais, arabe (syro-libainais du nord et du sud), farsi, ourdou, pachto |
| Références LDC | LDC2015E96 et LDC2015E97 |

que les quatre autres langues dans nos expériences avec ce corpus. Nous respectons le partage en ensembles d'apprentissage, de validation et de test des deux corpus d'origine. Les différents canaux sont approximativement équilibrés dans les corpus.

Notons qu'il existe un corpus *RATS LID* dédié à la reconnaissance de la langue (référence *LDC* LDC2018S10) mais que nous n'en disposions pas pour cette thèse.

## A.3 *Oriental Language Recognition* (*OLR*)

### A.3.1 Description des corpus d'origine

Grâce à notre participation à la compétition *Oriental Language Recognition 2020*, nous disposons des corpus des évaluations 2016 à 2020, décrits dans le tableau A.8.

### A.3.2 Constitution des corpus utilisés dans nos expériences

Lors de notre participation à la compétition (décrite en annexe D), nous avons réalisé avec ce corpus trois séries d'expériences : l'entraînement de *bottleneck features*, la sélection d'un modèle robuste et la conception du système final. Les corpus d'entraînement, de validation et de test associés à chaque série d'expériences sont décrits dans le tableau A.9. Lorsqu'un corpus d'origine est partagé en plusieurs corpus pour nos expériences, nous utilisons les annotations en locuteur (lorsqu'elles sont disponibles) pour assurer que des enregistrements d'un même locuteur ne soient pas répartis dans des corpus différents. Nous rapportons également quelques résultats sur les ensembles de test associés aux tâches 1 (canal inconnu) et 3 (environnement bruité) de l'édition 2020 de la compétition.

Pour l'entraînement de *bottleneck features*, nous définissions uniquement des ensembles d'entraînement et de validation.

Lors de la sélection d'un système robuste, nous utilisons uniquement les six langues présentes dans les corpus sur des canaux inconnus : japonais, mandarin, ouïghour, russe, tibétain et vietnamien. Nous définissons trois ensembles de test dont les noms correspondent aux corpus dont ils sont extraits : *test-2018*, *dev-2019* et *test-2019*. L'ensemble d'entraînement est appelé *entraînement mobile*.

La conception du système final est réalisée avec les seize langues des corpus *OLR*. Les deux ensembles de test, *test mobile* et *test mobile et canaux inconnus*, sont définis en fonction des canaux de transmission. L'ensemble d'entraînement est appelé *entraînement tous canaux*.

## A.4 Reconnaissance de la parole

Nous réalisons deux séries d'expériences avec des corpus de reconnaissance de la parole. Les corpus *Babel* sont utilisés pour développer des *bottleneck features*. Nous développement également des systèmes de reconnaissance de la parole pour la langue anglaise et utilisons certains corpus de référence pour cette tâche.

TABLE A.8 – Caractéristiques des corpus *Oriental Language Recognition*.

| Corpus | Langues | Canaux | Taille (durée et nombre de fichiers) |
|---|---|---|---|
| AP16-OL7 | cantonais, coréen, indonésien, japonais, mandarin, russe, vietnamien | téléphone mobile | 95 heures, 68430 |
| AP17-OL3 | kazakh, ouighour, tibétain | téléphone mobile | 46 heures, 26700 |
| AP18-OL7-test | cantonais, coréen, indonésien, japonais, mandarin, russe, vietnamien et huit langues inconnues | téléphone mobile | 24 heures, 18516 |
| AP18-OL3-test | kazakh, ouighour, tibétain | téléphone mobile | 10 heures, 5600 |
| AP19-dev | mandarin, japonais, kazakh, minnan, ouighour, russe, shanghaïen, tibétain, vietnamien et dialecte du Sichuan | téléphone mobile et canal inconnu | 5 heures, 4515 |
| AP19-test-tâche1 | cantonais, coréen, indonésien, japonais, kazakh, mandarin, ouighour, russe, tibétain, vietnamien | téléphone mobile | 5 heures, 18000 |
| AP19-test-tâche2 | japonais, mandarin, ouighour, russe, tibétain, vietnamien | canal inconnu | 10 heures, 10800 |
| AP19-test-tâche3 | minnan, shanghaïen et dialecte du Sichuan | téléphone mobile | 8 heures, 5430 |
| AP20-entraînement | minnan, shanghaïen et dialecte du Sichuan | téléphone mobile | 24000 |
| AP20-test-tâche1 | cantonais, coréen, indonésien, japonais, russe, vietnamien | canaux inconnux | 10800 |
| AP20-test-tâche2 | minnan, shanghaïen et dialecte du Sichuan | téléphone mobile | 5400 |
| AP20-test-tâche3 | cantonais, coréen, japonais, mandarin, russe | téléphone mobile, environnement bruité | 9000 |

TABLE A.9 – Utilisation des corpus pour nos trois séries d'expériences. Les abréviations *entr.* et *val.* signifient entraînement et validation

| Dataset | *bottleneck features* | *sélection d'un modèle robuste* | *conception du système final* |
|---|---|---|---|
| AP16-OL7 | entr. & val. | entr. & val. | entr. & val. |
| AP17-OL3 | entr. & val. | entr. & val. | entr. & val. |
| AP17-test | | entr. & val. | entr. & val. |
| AP18-test | | test-2018 | entr. & val. & test mobile |
| AP19-dev-tâche2 | | dev-2019 | test mobile et canaux inconnus |
| AP19-dev-tâche3 | | dev-2019 | entr. & val. |
| AP19 test tâche 1 | | test-2019 | entr. & val. & test mobile et canaux inconnus |
| AP19 test tâche 2 | | test-2019 | entr. & val. |
| AP19 test tâche 3 | | test-2019 | entr. & val. |
| AP20 entraînement | | | entr. & val. |

### A.4.1 Babel

#### Description des corpus d'origine

Les caractéristiques du corpus *Babel* sont rapportées dans le tableau A.10. Ce corpus est annoté en activité vocale et transcrit.

#### Constitution des corpus utilisés dans nos expériences

Lorsque nous entraînons des systèmes avec le corpus *Babel*, nous utilisons les quatorze langues suivantes : assamais, bengali, haïtien, kazakh, kurmandji, lao, lituanien, pachto, tamoul, télougou, tok pisin, turc, vietnamien, zoulou. Nous respectons les ensembles d'entraînement et de test définis par les corpus d'origine.

Nous utilisons également les modèles pré-entraînés de *multilingual features* produits par *BUT/Phonexia* [Silnova *et al.*, 2018] et entraînés sur dix-sept langues : les quatorze déjà citées ainsi que le cantonais, le cebuano et le tagalog.

### A.4.2 Langue anglaise

#### Description des corpus d'origine

**Librispeech** Le corpus *librispeech* [Panayotov *et al.*, 2015] est un corpus de livres audios lus à haute voix et enregistrés. Il compte près de mille heures d'enregistrement. Nous utilisons l'ensemble d'entraînement de cent heures appelé *train-clean*. Nous utilisons également des ensembles d'évaluation de référence, définis en fonction de la qualité du signal : *dev-clean*, *dev-other*, *test-clean* et *test-other*.

TABLE A.10 – Caractéristiques du corpus *Babel*

| | |
|---|---|
| Canal de transmission | téléphone |
| Durée | approximativement 200 heures par langue |
| Langues | amharique, assamais, bengali, cebuano, géorgien, guarani, haïtien, igbo, javanais, kazakh, kurmandji, lao, lituanien, luo, mongol, pachto, swahili, tagalog, tamoul, télougou, tok pisin, turc, vietnamien, zoulou |
| Références LDC | LDC2016S02, LDC2016S06, LDC2016S08, LDC2016S09, LDC2016S10, LDC2016S12, LDC2016S13, LDC2017S01, LDC2017S03, LDC2017S05, LDC2017S08, LDC2017S13, LDC2017S19, LDC2017S22, LDC2018S02, LDC2018S07, LDC2018S13, LDC2018S16, LDC2019S03, LDC2019S08, LDC2019S16, LDC2019S22, LDC2020S02, LDC2020S07, LDC2020S10 |

**Switchboard et CallHome** Les corpus *switchboard* [Godfrey *et al.*, 1992] et *callhome* correspondent à des conversations téléphoniques. Nous utilisons une partie du corpus *switchboard* pour l'entraînement de systèmes, tandis que les deux corpus sont utilisés pour l'évaluation avec le corpus appelé *eval2000* [Fiscus *et al.*, 2000] qui contient des enregistrements de *callhome* et de *switchboard*.

**CommonVoice** Le corpus *common voice* [Ardila *et al.*, 2020] est un corpus reposant sur un site *web* qui permet à des volontaires de s'enregistrer afin de constituer un corpus ouvert pour le développement d'applications de traitement de la parole. En 2020, plus de cinquante mille locuteurs ont participé, ce qui résulte en plus de deux mille cinq cents heures de parole en trente-huit langues.

Dans notre travail, nous nous limitons à l'utilisation des enregistrements anglais dans le but de tester les variabilités dues au genre et à l'accent. Nous utilisons en particulier des enregistrements de locuteurs américains (États-Unis), anglais et indiens.

### Constitution des corpus utilisés dans nos expériences

Dans nos expériences, nous utilisons deux corpus d'entraînement : le corpus d'entraînement canonique de cent heures de *librispeech* et un sous-ensemble de *switchboard* de cent heures. Nous évaluons les modèles sur les ensembles de test canoniques de *librispeech* ainsi que sur le corpus *eval2000* contenant des enregistrements de *switchboard* et *callhome*. *Common voice* est exclusivement utilisé pour l'évaluation des modèles en définissant des sous-ensembles particuliers en fonction de l'accent et du genre des locuteurs.

## A.5 Corpus utiles aux augmentations de données

### A.5.1 *MUSAN : A Music, Speech, and Noise Corpus*

Le corpus *MUSAN* [Snyder *et al.*, 2015] est utilisé pour réaliser des augmentations de données, en ajoutant des bruits additifs. Il contient cent neuf heures d'audio, échantillonné à 16 kHz. Trois types d'enregistrements sont fournis. Une première partie correspond à soixante heures de parole, en anglais et en onze autres langues. Ces enregistrements proviennent du corpus *Librivox* (livres audio) ou d'enregistrements du gouvernement américain (débats, auditions, travaux en commission). Une seconde partie de quarante-deux heures correspond à des enregistrements musicaux annotés en genre, notamment baroque, classique, romantique, *jazz*, *bluegrass*, *hiphop*. Enfin une dernière partie de six heures correspond à des bruits, par exemple bruits de répondeur, d'imprimante, bruits de voiture, de vent, de froissement de papier, de pluie, cris d'animaux mais aussi des bruits de foule avec des voix inintelligibles.

### A.5.2 *Room Impulse Response* (*RIR*)

Le corpus *RIR* [Ko *et al.*, 2017] est un corpus de réponses impulsionnelles de pièces. Il comprend des réponses réelles, des bruits isotropiques, ainsi que des réponses impulsionnelles simulées et des sources ponctuelles extraites du corpus *MUSAN*.

# B

# Résultats complémentaires sur le corpus *OpenSAD 15*

Nous rapportons ici les résultats d'expériences complémentaires à la section 4.3, réalisées sur le corpus *OpenSAD 15* qui compte quatre langues et sept canaux de transmission.

Nous présentons d'abord la perte de performance du système *CNN* (tableau 4.8) lorsqu'il est évalué sur un canal de transmission différent du canal d'entraînement. Le système *CNN* est central dans notre étude puisque nous l'avons utilisé dans les expériences du chapitre 4 pour déterminer les hyper-paramètres de la méthode de régularisation d'un réseau de neurones.

D'autre part, nous présentons ici certaines des expériences réalisées avec les systèmes *CNN* et *x-vectors* dans le chapitre 4 pour un scénario d'adaptation de domaine non supervisée, pour un troisième système que nous nommons *BNF-CNN*. Ces expériences donnent des résultats cohérents avec l'analyse présentée dans la section 4.3 avec les deux autres systèmes.

## B.1 Perte de performance sur des canaux inconnus du système *CNN*

Le système *CNN* est décrit dans le tableau 4.8. Il est constitué d'un réseau de neurones convolutionnel qui réalise directement la tâche de reconnaissance de la langue à partir de *features* acoustiques. Nous entraînons un système sur chacun des canaux de transmission du corpus *OpenSAD 15* et l'évaluons sur chaque canal. Finalement, un dernier système est entraîné avec un corpus d'entraînement contenant l'ensemble des canaux de transmission.

Les performances rapportées dans le tableau B.1 illustrent le comportement déjà observé dans la section 4.1.1. Le système *CNN* subit une perte de performance lorsqu'il est évalué sur un canal de transmission différent du canal d'entraînement. S'il est entraîné sur l'ensemble des canaux, il atteint une bonne performance sur chacun d'entre eux.

TABLE B.1 – Performance du système *CNN* entraîné pour quatre langues sur le corpus *OpenSAD 15* en fonction des canaux d'entraînement et de test. $EER_{avg}$ (%).

| | Canal de test | | | | | | |
|---|---|---|---|---|---|---|---|
| Canal d'entraînement | src | B | D | E | F | G | H |
| src (téléphone) | **8** | 57 | 52 | 48 | 51 | **30** | 50 |
| B (UHF) | 52 | **18** | 57 | 56 | 59 | 53 | 61 |
| D (HF) | 48 | 57 | **15** | 57 | 56 | 53 | 61 |
| E (VHF) | 46 | 69 | 63 | **19** | 54 | 60 | 38 |
| F (UHF) | 33 | 53 | 60 | 44 | **15** | 47 | 53 |
| G (UHF) | **19** | 51 | 48 | 45 | 54 | **14** | 48 |
| H (HF) | 48 | 63 | 51 | 40 | 56 | 53 | **22** |
| src - B - D - E - F - G - H | **11** | **21** | **12** | **19** | **15** | **15** | **20** |

## B.2 Adaptation de domaine non supervisée avec le système *BNF-CNN*

Le système *BNF-CNN* est constitué d'un extracteur de *bottleneck features* et d'un réseau de neurones convolutionnel et est décrit dans le tableau B.2. Il est également entraîné et évalué sur le corpus *OpenSAD 15*.

Nous vérifions en premier lieu que le système *BNF-CNN* subit une perte de performance lorsqu'il est évalué sur un canal de transmission différent du canal d'entraînement. Les performances de cette série d'expériences sont présentées dans le tableau B.3. Les résultats sont similaires à ceux observés avec le système *CNN* (section B.1) et le système *x-vector* (section 4.1.1).

Ensuite, nous comparons les trois fonctions de régularisation de distance entre les moyennes, *CORAL* et *MMD* en les appliquant sur la couche de sortie du réseau convolutionnel, dans le but de réaliser une adaptation de domaine non supervisée. Le domaine source est le canal téléphonique (*src*) et des systèmes sont entraînés pour chaque canal radio comme domaine cible. Nous utilisons les hyper-paramètres sélectionnés par les expériences avec le système *CNN* avec les canaux G et D. Les résultats présentés dans le tableau B.4 montrent que les trois fonctions de régularisation améliorent la performance sur le domaine cible. La plus performante est la *MMD* qui atteint une performance du même ordre qu'un entraînement supervisé sur le domaine cible. Ces résultats sont similaires à ceux de la section 4.3.5.

Enfin, nous réalisons une adaptation de domaine non supervisée vers les six canaux cibles simultanément en ajoutant les fonctions de régularisation pour chaque canal cible. Nous vérifions dans le tableau B.5 qu'une telle adaptation simultanée est possible. Ces résultats sont comparables à ceux présentés dans la section 4.3.6.

TABLE B.2 – Architecture du système *BNF-CNN*

| Prédiction directe de scores d'identification de la langue par un réseau convolutionnel prenant en entrée des *bottleneck features* multilingues, calculés à partir de spectrogrammes en échelle *Mel*. | | |
|---|---|---|
| Module | Modèle | Hyper-paramètres |
| détection d'activité vocale | annotation manuelle | - |
| *features* acoustiques | log-spectrogrammes en échelle *Mel* | $\Delta_F = 25ms$, $\delta_F = 10ms$, prétraitement décrit dans [Silnova *et al.*, 2018] |
| augmentations de données | non utilisés | - |
| *bottleneck features* | *stacked bottleneck features* | modèle multilingue produisant des *features* de dimension 80. Nous utilisons le modèle de *BUT / Phonexia* [Silnova *et al.*, 2018] entraîné à prédire des triphones pour 17 langues du corpus *Babel*. |
| modélisation du segment | réseau de neurones convolutionnel (tableau 4.7) | entraîné pour l'identification de la langue avec l'entropie croisée |
| classifieur final | non utilisé | utilisation directe des probabilités *a posteriori* produites par le réseau |
| calibration | non utilisée | - |

TABLE B.3 – Performance du système *BNF-CNN* entraîné pour quatre langues sur le corpus *OpenSAD 15* en fonction des canaux d'entraînement et de test. $EER_{avg}$ (%).

| | Canal de test | | | | | | |
|---|---|---|---|---|---|---|---|
| Canal d'entraînement | src | B | D | E | F | G | H |
| src (téléphone) | **3** | 37 | 42 | 33 | 30 | **6** | 34 |
| B (UHF) | 30 | **8** | 27 | 32 | 45 | 31 | 34 |
| D (HF) | 41 | 36 | **4** | 37 | 42 | 41 | 40 |
| E (VHF) | 29 | 30 | 41 | **9** | 37 | 30 | 28 |
| F (UHF) | 34 | 41 | 33 | 42 | **8** | 33 | 35 |
| G (UHF) | **5** | 29 | 39 | 35 | 32 | **3** | 32 |
| H (HF) | 30 | 37 | 33 | 26 | 40 | 27 | **9** |
| src - B - D - E - F - G - H | **3** | **9** | **5** | **9** | **6** | **4** | **9** |

TABLE B.4 – Performance en terme de moyenne des taux d'égale erreur de différentes méthodes d'entraînement du réseau convolutionnel du système *BNF-CNN* pour le corpus *OpenSAD 15* (segments de trois secondes). Le domaine source est le canal téléphonique (*src*).

| Méthode d'apprentissage | $EER_{avg}$ sur le domaine cible (%) | | | | | |
|---|---|---|---|---|---|---|
| | *B* | *D* | *E* | *F* | *G* | *H* |
| supervisé sur *source* | 37 | 42 | 33 | 30 | 6 | 34 |
| supervisé sur *cible* | **8** | **4** | **9** | 8 | **3** | **9** |
| distance entre les moyennes | 29 | 23 | 27 | 28 | **3** | 22 |
| *CORAL* | 26 | 6 | 21 | 15 | 4 | 31 |
| *MMD* | 11 | 6 | **9** | **7** | **3** | 10 |

TABLE B.5 – Performance en terme de moyenne des taux d'égale erreur de différentes méthodes d'entraînement du réseau convolutionnel du système *BNF-CNN* pour le corpus *OpenSAD 15* (segments de trois secondes). Le domaine source est le canal téléphonique (*src*).

| Méthode d'apprentissage | $EER_{avg}$ sur le domaine cible (%) | | | | | |
|---|---|---|---|---|---|---|
| | *B* | *D* | *E* | *F* | *G* | *H* |
| supervisé sur *source* | 37 | 42 | 33 | 30 | 6 | 34 |
| supervisé sur chaque *cible* séparément | **8** | **4** | **9** | 8 | **3** | **9** |
| supervisé sur 7 canaux simultanément | 9 | 5 | **9** | **6** | 4 | **9** |
| MMD par canal | 11 | 6 | **9** | 7 | **3** | 10 |
| MMD simultanée vers 6 canaux | 11 | 7 | **9** | 7 | 4 | 11 |

# C

# Système de détection d'activité vocale

Dans certains systèmes de reconnaissance de la langue, nous avons utilisé un système de détection de la parole fondé sur un réseau de neurones convolutionnel. Nous décrivons ici ce système.

La tâche de détection de la parole consiste à produire une segmentation du signal en segments de parole et de non parole. Un système réalisant cette tâche est utilisé afin de filtrer les trames de *features* fournies en entrée du réseau d'identification de la langue au sein d'un système de reconnaissance de la langue.

## C.1 Description du système

Le système contient trois modules. D'abord des *features* acoustiques représentant le signal sont extraits. Ensuite, un réseau de neurone convolutionnel prédit un score de détection de parole par trame. Finalement, une segmentation du signal est produite par un module de décision qui prend en entrée la séquence des scores.

Les *features* acoustiques sont des *MFCC*. Ils sont produits pour des trames de 25 ms avec un pas de 10 ms. 24 coefficients en échelle *Mel* sont extraits sur une gamme de fréquence entre 0 et 4000 Hz, 13 coefficients *MFCC* sont conservés. Une normalisation des moyennes des *features* sur l'ensemble du segment est appliquée.

La prédiction d'un score par trame est assurée par un réseau de neurones convolutionnel. Les couches convolutives sont des convolutions à une dimension. Le réseau prend en entrée 21 trames : en plus de la trame centrale, des contextes gauche et droit de dix trames sont utilisés. Ces trames sont traitées par trois couches convolutives et les vecteurs obtenus sont concaténés puis traités par une couche connectée pour produire un score de détection de la parole. Ce réseau est entraîné à minimiser l'entropie croisée sur le corpus *RATS SAD* (annexe A.2) par une descente de gradient stochastique. Une seule époque d'entraînement est réalisée, en utilisant les neuf canaux de transmission du corpus.

La segmentation finale est réalisée avec une version simplifiée de la recette exposée dans [Gelly, 2017] dans le but d'éviter les faux rejets. Le seuil est appliqué aux prédictions

du réseau de neurones pour détecter les trames de parole. Ensuite, tous les segments de non parole contenus entre deux trames de parole et de durée inférieure à 200 ms sont prédits comme de la parole. Ensuite, tous les segments de parole de durée inférieure à 150 ms sont éliminés. Finalement, 100 ms de sécurité sont ajoutées au début et à la fin de chaque segment de parole.

## C.2 Performance

Nous évaluons la performance du système entraîné en terme de détection de trames de parole. Les résultats présentés dans le tableau C.1 sont des taux d'égale erreur calculés par canal. Pour chaque canal, le seuil correspondant est rapporté. Un taux d'égal erreur et le seuil associé sont également calculés pour l'ensemble des canaux. C'est ce seuil que nous utilisons dans nos systèmes de reconnaissance de la langue. Il est particulièrement conservatif pour le canal téléphonique (*src*).

TABLE C.1 – Performance du système de détection d'activité vocale sur l'ensemble de test du corpus *RATS SAD*. Taux d'égale erreur (%) et seuil associé.

| | Canal de test | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | src | A | B | C | D | E | F | G | H | tous |
| $EER$ (%) | 5,4 | 9,2 | 9,2 | 9,1 | 15,6 | 7,7 | 8,4 | 7,8 | 7,5 | 8,9 |
| seuil | 0,585 | 0,376 | 0,289 | 0,349 | 0,229 | 0,593 | 0,342 | 0,398 | 0,255 | 0,355 |

# D

# Participation à la compétition *Oriental Language Recognition 2020*

La compétition *Oriental Language Recognition* (reconnaissance de langues orientales en français) est organisée chaque année depuis 2016 dans le but de soutenir et d'orienter la recherche sur la reconnaissance de la langue [Wang *et al.*, 2016, Tang *et al.*, 2017a, Zhiyuan Tang, 2018, Tang *et al.*, 2019]. Nous avons présenté cette série de campagnes d'évaluation dans la section 2.1.7. L'édition 2020 [Li *et al.*, 2020a] proposait trois tâches : la reconnaissance de la langue sur un canal inconnu (tâche 1), l'identification de dialectes (tâche 2) et l'identification de la langue dans des conditions bruitées (tâche 3). Nous avons décidé de concentrer notre effort de recherche sur la généralisation à des conditions inconnues dans le but de participer aux tâches 1 et 3. Nous n'avons pas participé à la tâche 2.

Certaines expériences réalisées durant le travail pour cette compétition ont été présentées dans ce document : la recette d'entraînement de l'extracteur de *bottleneck features* à partir d'un modèle *Conformer* et la comparaison de certains systèmes sur les corpus de la compétition. Nous décrivons ici l'ensemble des étapes du développement du système soumis de façon cohérente, comme un exemple de la mise en œuvre des approches proposées dans cette thèse dans le but d'optimiser la performance pour une tâche particulière. Cette annexe reprend les résultats exposés dans le papier décrivant notre système [Duroselle *et al.*, 2021a]. La bonne performance obtenue lors de la compétition (première place pour les deux tâches pour lesquelles nous avons concouru) valide les méthodes d'augmentation de la robustesse au canal que nous avons étudiées.

Nous avons mené trois séries d'expériences dans le but de développer un système robuste de reconnaissance de la langue. Nous avons d'abord développé un extracteur de *bottleneck features* robuste en utilisant un modèle de reconnaissance de la parole de séquence à séquence, *Conformer*. Ensuite, dans le but de simuler une évaluation sur des conditions inconnues, nous avons comparé différentes recettes d'entraînement en utilisant un corpus d'entraînement constitué uniquement de données téléphoniques et en évaluant

la performance avec des canaux de transmission inconnus. Nous appelons cette étape la *sélection d'un modèle robuste*. Enfin, nous avons combiné plusieurs systèmes, entraînés sur un ensemble d'entraînement plus grand incluant des donnés de canaux inconnus dans le but d'augmenter la robustesse. Nous appelons cette étape la *conception du système final*. Des ensembles d'entraînement et d'évaluation spécifiques ont été définis pour chaque série d'expériences.

Le système soumis est le même pour les tâches 1 et 3. Il est constitué de la combinaison de cinq modèles : un *GMM* et quatre *TDNN* entraînés avec des *bottleneck features* et différentes fonctions de coût. La bonne généralisation a été possible grâce à la nouvelle recette d'entraînement de *bottleneck features*, à des augmentations de données, à la combinaison de modèles aux propriétés complémentaires et à la stratégie de sélection du modèle final *stochastic weight averaging* [Izmailov *et al.*, 2018]. Nous avons également proposé une calibration dépendant de la durée des segments, sans que cela améliore la performance du système.

## D.1 Utilisation des données

### D.1.1 Corpus

Les règles de la compétition spécifient un ensemble limité de corpus pouvant être utilisés pour développer les systèmes. Ils sont listés dans le tableau A.9. L'ensemble d'entraînement compte seize langues. Les dix langues dites *traditionnelles* de la compétition sont le cantonais (ct-cn), le mandarin (zh-cn), l'indonésien (id-id), le japonais (ja-jp), le russe (ru-ru), le coréen (ko-kr), le vietnamien (vi-vn), le kazakh (ka-cn), le tibétain (ti-cn) et le ouïghour (uy-cn). Seules ces langues sont fournies avec des transcriptions dans les ensembles AP16-OL7 et AP17-OL3. Des enregistrements de six autres langues sont fournies : trois dialectes chinois, le minnan, un dialecte du Sichuan et le shangaïen, et trois langues non cibles des évaluations précédentes, catalan, grec et télougou. La composition des corpus est détaillée en annexe A.3.

La majorité des corpus est constituée d'enregistrements de téléphones mobiles. Seuls deux corpus correspondent à des canaux de transmission inconnus : AP19-dev-tâche2 et AP19-test-tâche2. Ils ne contiennent que six langues : japonais, russe, vietnamien, tibétain, mandarin et ouïghour. Nous les appelons les *langues de développement*. Notons que nous n'avons pas accès à des canaux de transmission autres que téléphonique pour trois des six langues cibles de la tâche 1 : cantonais, indonésien et coréen. Par conséquent, nous avons défini une stratégie en deux étapes pour utiliser au mieux ces données de canaux de transmission inconnus. D'abord nous les avons gardées pour évaluer les modèles sur des canaux inconnus (*sélection d'un modèle robuste*). Ensuite nous les avons ajoutées à l'ensemble d'entraînement pour en augmenter la diversité (*conception du système final*).

Le système soumis ayant été conçu après trois séries d'expériences, des ensembles d'entraînement, de validation et de test ont été définis pour chaque étape. Ces choix sont décrits dans le tableau A.9. Les trois séries d'expériences sont les suivantes :

1. Entraînement d'un extracteur de *bottleneck features*. Nous utilisons des enregistre-

ments des dix *langues traditionnelles* avec leurs transcriptions.

2. *Sélection d'un modèle robuste.* Dans le but de simuler une évaluation sur canal de transmission inconnu, nous entraînons des systèmes en utilisant uniquement des enregistrements de téléphones mobiles. Nous évaluons les systèmes sur trois corpus de test : *test-2018* (uniquement des données téléphoniques), *dev-2019* et *test-2019* (avec des données de canaux inconnus). Pour nous concentrer sur le problème du changement de canal de transmission, nous utilisons uniquement les six *langues de développement*.
3. *Conception du système final.* Une fois que nous avons sélectionné une recette d'entraînement robuste au changement de canal de transmission, nous augmentons la diversité de l'ensemble d'entraînement en ajoutant des données de canaux inconnus. Nous entraînons les systèmes avec les seize langues du corpus. Cela nous autorise à les évaluer pour trois ensembles de langues différents : les *langues de développement*, les langues de la tâche 1 et les langues de la tâche 3. Nous utilisons deux ensembles de test : *test mobile* (avec du téléphone mobile uniquement) et *test mobile et canaux inconnus* (avec certains enregistrements de canaux inconnus).

### D.1.2 Augmentations de données

Pour augmenter la robustesse des systèmes à des conditions inconnues, nous les avons entraînés avec trois méthodes d'augmentation de données. Conformément aux règles de la compétition [Li *et al.*, 2020a], nous n'avons pas utilisé de ressources supplémentaires et nous sommes limités aux autres fichiers du corpus d'entraînement ou à la génération de bruits ou de filtres artificiels. Les trois méthodes d'augmentation de données sont : l'ajout de bruit blanc additif, l'ajout de *babble noise* (en mélangeant d'autres fichiers du corpus d'entraînement) et la convolution avec des filtres passe-bande aléatoires.

## D.2 *Features* à l'échelle de la trame

Nous avons exploré deux types de *features* : des représentations spectrales et des *bottleneck features* (*BNF*). Nous avons également utilisé des *SDC-MFCC* spécifiquement pour le *GMM*.

### D.2.1 Représentations spectrales

Des spectrogrammes en échelle *Mel* ainsi que des *MFCC* ont été évalués. Quand ils sont utilisés sans méthode d'augmentation de la robustesse, la restriction à la bande téléphonique (300-3800 Hz) permet d'en améliorer la robustesse. Aucune amélioration significative n'a été obtenue en ajoutant des *features* caractérisant le *pitch* [Talkin et Kleijn, 1995].

Le modèle de mélange de Gaussiennes (*GMM*) emploie des *features SDC-MFCC* [Torres-Carrasquillo *et al.*, 2002]. Les *MFCC* sont calculés pour des trames de 20 ms avec un pas de 10 ms et 16 filtres en échelle *Mel*. Les *MFCC* extraits sont traités par la

méthode *RASTA* pour réduire la variation due au canal. Ensuite, des *SDC* sont calculés avec un décalage de trois trames et un contexte de sept pour créer des *SDC-MDCC* de dimension 128.

Pour tous les sytèmes, nous avons appliqué un algorithme de détection de la parole basé sur l'énergie du signal. Enfin, la normalisation de la moyenne et de la variance des représentations cepstrales (*CMVN*) a permis d'améliorer la robustesse des *TDNN*.

### D.2.2 *Bottleneck features*

C'est au cours de la compétition *OLR 2020* que nous avons développé l'extraction de *bottleneck features* depuis un modèle de reconnaissance de la parole de bout en bout multilingue. L'architecture du modèle et la méthode d'entraînement ont été exposées dans la partie 6.2 tandis que les détails d'implémentation pour le corpus *OLR 2020* ont été donnés dans la section 6.2.3

## D.3 Entraînement des réseaux de neurones d'identification de la langue

Dans quatre des cinq sous-systèmes faisant partie du système soumis, la classification à l'échelle du segment est réalisée par un réseau de neurones. Il s'agit du *TDNN* standard pour la reconnaissance de la langue [Snyder *et al.*, 2018]. Nous avons d'abord sélectionné la recette d'entraînement générale avec les corpus définis pour la *sélection d'un modèle robuste*. Ensuite, nous avons entraîné plusieurs systèmes avec les données de *conception du système final*, en utilisant des fonctions de coût qui exploitent la présence de données de canaux de transmission inconnus dans l'ensemble d'entraînement.

### D.3.1 Recette d'entraînement

La recette d'entraînement sélectionnée comprend les éléments suivants :

— une fonction de coût de classification, entropie croisée (*CE*) ou *additive angular margin softmax* (*AAM*) [Deng *et al.*, 2019]
— une descente de gradient stochastique avec *dropout* [Srivastava *et al.*, 2014]
— la régularisation *specAugment* [Park *et al.*, 2019]
— les trois méthodes d'augmentation de données décrites dans la section D.1.2
— la sélection du modèle final avec la stratégie *stochastic weight averaging* [Izmailov *et al.*, 2018]

### D.3.2 Exploration d'autres fonctions de coût

Pour la dernière série d'expériences, les corpus de *conception du système final* contenaient des enregistrements de canaux de transmission inconnus. Par conséquent nous avons utilisé des fonctions de coût qui visent à réduire l'écart entre les représentations associées aux différents canaux de transmission :

— régularisation de l'entropie croisée avec la *MMD* entre les enregistrements du canal téléphonique et les canaux inconnus, nous avons comparé plusieurs hyperparamètres de poids $\lambda$ de la régularisation
— régularisation de l'entropie croisée avec la *n-pair loss*

## D.4 Entraînement du *GMM*

Dans le but de combiner des sous-systèmes aux propriétés différentes, nous avons entraîné un *GMM* à réaliser la tâche de reconnaissance de la langue. Nous avons entraîné un mélange de Gaussiennes avec 4096 composantes pour chaque langue avec 10 itérations de l'algorithme *expectation-maximization* (*EM*). Ce modèle produit une vraisemblance par trame et nous moyennons ensuite les scores pour l'ensemble des trames du segment.

## D.5 Fusion et calibration

La fusion des scores des sous-systèmes et la calibration des scores a été réalisée à l'aide de l'outil *FoCal* [Brümmer, 2007], avec l'ensemble de validation défini pour l'étape de *conception du système final*. Les scores produits par le système sont des logarithmes de rapports de vraisemblance.

Nous avons appliqué une calibration dépendant de la durée des segments de test au système final [McLaren *et al.*, 2018b]. Nous avons utilisé trois intervalles de durée : inférieur à deux secondes, de deux à quatre secondes et supérieur à quatre secondes.

## D.6 Expériences

Dans cette section, nous exposons les principaux résultats ayant mené à la conception du système soumis.

### D.6.1 Entraînement de l'extracteur de *bottleneck features*

Cette étape utilise les corpus *bottleneck features*. Les modèles sont comparés avec la *loss CTC* pour les dix langues cibles sur l'ensemble de validation dans le tableau D.1. Il apparaît que la combinaison de la normalisation des *features* (*CMVN*), de la régularisation *specAugment* et des augmentations de données permet d'améliorer la performance de reonnaissance de la parole. Les *features* obtenus par l'application de ces trois méthodes de régularisation sont appelés *BNF finaux* par opposition aux *BNF de base* et utilisés pour le système soumis.

### D.6.2 Sélection d'un modèle robuste

Pour cette série d'expériences, nous entraînons des modèles d'identification de la langue avec un ensemble d'entraînement contenant uniquement des enregistrements de téléphones mobiles. La performance est mesurée avec le taux d'égal erreur (*EER*) sur trois

TABLE D.1 – Performance de reconnaissance de la parole en terme de moyenne de la *CTC loss* sur l'ensemble de validation pour différentes recettes d'entraînement. Une méthode de régularisation supplémentaire est ajoutée par ligne du tableau, autrement dit la dernière ligne utilise toutes les méthodes de régularisation.

| Nom du système | Recette d'entraînement | *CTC loss* sur l'ensembme de validation |
|---|---|---|
| *BNF de base* | spectrogrammes en échelle *Mel* | 3,94 |
| | + *CMVN* | 3,69 |
| | + *specAugment* | 3,52 |
| *BNF finaux* | + augmentations de données | **3,02** |

ensembles d'évaluation : *test-2018* (téléphone mobile), *dev-2019* et *test-2019* (téléphone mobile et canaux inconnus), avec les six *langues de développement*, cf. tableau D.2. Nous utilisons un *TDNN* entraîné avec la recette décrite en section D.3.1 avec différents *features* d'entrée et deux méthodes de sélection du modèle final.

TABLE D.2 – Performance de reconnaissance de la langue pour les expériences de *sélection d'un modèle robuste*. Moyenne des taux d'égale erreur ($EER_{avg}$, %) sur les ensembles d'évaluation. Tous les modèles sont des *TDNN* entraînés avec l'entropie croisée et les augmentations de données.

| *features* d'entrée | méthode de sélection | test-2018 | dev-2019 | test-2019 |
|---|---|---|---|---|
| spectrogrammes en échelle *Mel* | coût min. sur validation | 11,95 | 25,48 | 29,54 |
| *MFCC* | coût min. sur validation | 4,75 | 37,78 | 24,34 |
| *BNF de base* | coût min. sur validation | 3,91 | 19,11 | **11,08** |
| *BNF finaux* | coût min. sur validation | 3,43 | 17,24 | 13,68 |
| *BNF finaux* | *stochastic weight averaging* | **2,97** | **16,92** | 12,78 |

Les expériences montrent la supériorité des *bottleneck features* sur les spectrogrammes en échelle *Mel* et les *MFCC*. Les *BNF finaux* sont meilleurs que les *BNF de base* pour l'ensemble d'évaluation le plus difficile, *dev-2019*. Finalement la méthode *stochastic weight averaging* est supérieure à la sélection du modèle minimisant le coût sur l'ensemble de validation. Cette stratégie a donc été appliquée à tous les systèmes soumis.

### D.6.3 Conception du système final

Nous avons entraîné plusieurs modèles avec un grand ensemble d'entraînement contenant des enregistrements de seize langues et certains canaux de transmission inconnus. Nous les avons calibrés sur l'ensemble de validation et évalués sur deux corpus en terme de $C_{avg}$ (la métrique de la compétition) : *test mobile* et *test mobile et canaux inconnus*. La performance est calculée pour deux ensembles de langues cibles correspondant aux tâches 1 et 3. Les résultats sont présentés dans le tableau D.3.

Pour montrer l'apport de l'entraînement sur un corpus plus grand, nous rapportons la performance obtenue par le modèle *TDNN* entraîné sur le corpus de *sélection d'un*

TABLE D.3 – Performance pour les expériences de *conception du système final* en termes de $C_{avg}$. La performance est mesurée pour les trois ensembles de développement, *validation* (val.), *test mobile* (mob.) et *test mobile et canaux inconnus* (can.) ainsi que les ensembles d'évaluation de la compétition (eval). Le système soumis correspond à la dernière ligne du tableau.

| | modèle | Tâche 1 (six langues) $C_{avg} \times 100$ | | | | Tâche 3 (cinq langues) $C_{avg} \times 100$ | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | val. | mob. | can. | eval | val. | mob. | can. | eval |
| modèles individuels | *baseline pytorch x-vector* [Li *et al.*, 2020a] | | | | 13,21 | | | | 7,15 |
| | CE, *sélection d'un modèle robuste* | 3,94 | 5,60 | 10,30 | **4,96** | 3,48 | 5,91 | 9,25 | **4,02** |
| | CE, *conception du système final* | **2,50** | 4,69 | 7,26 | 5,12 | **2,04** | 4,34 | 7,16 | **4,94** |
| | **AAM** | 3,68 | 3,14 | **5,34** | **5,08** | 3,87 | 3,20 | **6,25** | 5,11 |
| | *n-pair* | 3,22 | 5,73 | 7,25 | 7,52 | 2,58 | 5,14 | 7,36 | 5,70 |
| | MMD $\lambda = 1$ | 3,18 | 5,39 | 7,87 | 5,54 | 2,93 | 4,99 | 8,31 | 5,58 |
| | MMD $\lambda = 100$ | 4,01 | 5,89 | 7,60 | 7,61 | 3,76 | 5,24 | 7,29 | 5,80 |
| | GMM | 3,31 | **2,53** | 10,82 | 7,13 | 3,00 | **2,69** | 13,97 | 9,32 |
| fusion | AAM-GMM | 1,37 | 1,54 | 4,37 | 2,82 | 1,35 | **1,45** | 5,73 | 4,93 |
| | AAM-GMM-MMD100 | 0,98 | 1,57 | 3,91 | 2,47 | 1,00 | 1,60 | 4,96 | 3,82 |
| | **AAM-GMM-MMD100-*n-pair*-MMD1** | **0,93** | **1,53** | **3,67** | **2,28** | **0,97** | 1,60 | **4,54** | **3,71** |
| | **calibration dépendant de la durée** | | 1,56 | 3,82 | 2,39 | | 1,61 | 4,60 | 3,74 |

*modèle robuste*. Tous les autres modèles sont entraînés avec les corpus de *sélection du système final* et atteignent une meilleure performance sur les corpus de développement. Parmi les modèles individuels, la meilleure performance pour des canaux de transmission inconnus est obtenue par le *TDNN* entraîné avec la fonction de coût *AAM*. Les modèles entraînés avec des fonctions de coût de régularisation (*MMD* et *n-pair loss*) n'obtiennent pas une meilleure performance mais l'écart entre les canaux connus et inconnus est réduit. Enfin, le *GMM* atteint la meilleure performance sur le canal mobile mais est très sensible à l'écart entre les canaux.

En partant du modèle obtenant la meilleure performance, nous ajoutons successivement des modèles au système final. Nous sélectionnons la combinaison de cinq systèmes : quatre *TDNN* utilisant des *bottleneck features* et entraînés avec la fonction de coût *AAM*, la régularisation avec la *MMD* avec un poids $\lambda$ fort et faible, la régularisation avec la *n-pair loss* et un *GMM*.

Sur le canal mobile, le meilleur système individuel est le *GMM* ($C_{avg} \times 100 = 2{,}53$ pour la tâche 1). La combinaison avec le *TDNN* entraîné avec la fonction de coût *AAM* permet d'améliorer la performance (1,54) mais l'ajout de trois autres *TDNN* entraînés avec des fonctions de coût visant à augmenter la robustesse n'octroie pas un gain supplémentaire (1,53). À l'inverse, pour des canaux de transmission inconnus (colonnes *can.* dans le tableau D.3), le *TDNN* entraîné avec *AAM* est meilleur que le *GMM* (5,34 au lieu de 10,82 pour la tâche 1). La fusion des deux est utile et la combinaison avec trois autres modèles robustes permet une amélioration nette de la performance (3,67). Même si les modèles entraînés avec des fonctions de coût de régularisation atteignent une performance individuelle assez faible, ils sont conçus pour être robustes au changement de canal et

sont donc utiles lors de la fusion avec des modèles plus performants mais moins robustes.

Enfin, nous apprenons des paramètres de fusion et de calibration spécifiques pour chaque intervalle de durées. Lors de l'inférence, nous utilisons les paramètres correspondant à la durée de parole du segment de test. Comme nous n'avions pas d'information sur les durées des segments de l'ensemble d'évaluation, nous avons choisi d'utiliser cette stratégie qui nous semblait plus pertinente, même si elle ne permettait pas d'améliorer la performance sur nos ensembles de développement.

## D.7 Analyse du système soumis

Le système soumis correspond à la dernière ligne du tableau D.3. Il a atteint la meilleure performance pour les tâches 1 et 3 de la compétition. Les taux d'égale erreur correspondant sur les ensembles d'évaluation sont 2.47% pour la tâche 1 et 4.07% pour la tâche 3.

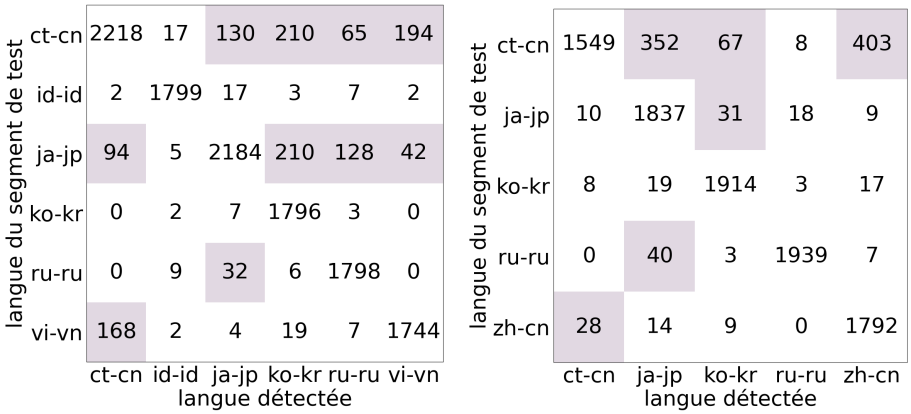
Après la révélation des étiquettes des ensembles d'évaluation, nous avons analysé la performance de nos systèmes sur ces ensembles. Les performances des systèmes sur les ensembles d'évaluation sont cohérentes avec les performance sur l'ensemble *test mobile et canaux inconnus* (voir tableau D.3), à l'exception des modèles entraînés avec l'entropie croisée qui obtiennent une bien meilleure performance sur les ensembles d'évaluation. Nous n'avons pas d'interprétation claire de cette différence de comportement. La calibration dépendant de la durée n'a pas permis d'améliorer la performance.

Pour les deux tâches, nous avons analysé les erreurs du système soumis en fonction de la durée des segments et d'estimations du rapport signal à bruit et de la fréquence fondamentale des locuteurs (comme approximation du genre). Sans grande surprise, nous avons vérifié que la performance du système soumis s'améliorait pour des durées plus longues et des niveaux de bruit plus faibles. D'autre part, le système est plus performant pour des fréquence fondamentales hautes (200-250 Hz) que basses (100-150 Hz).

Finalement, pour les deux tâches les erreurs sont dominées par quelques paires de langues. Ce phénomène peut être observé sur les matrices de confusion de la figure D.1. Une amélioration en terme de $C_{avg}$ peut être espérée si nous nous concentrons sur ces paires spécifiques.

## D.8 Conclusion

La compétition *OLR 2020* a été l'occasion de mettre en oeuvre les méthodes développées au cours de cette thèse pour un scénario d'apprentissage particulièrement difficile : la généralisation à un canal de transmission inconnu. Dans ce cadre, nous avons montré l'apport de la recette d'entraînement de *bottleneck features* à partir d'un modèle de reconnaissance de la parole *end-to-end* multilingue et l'intérêt des méthodes de régularisation de la fonction de coût du réseau d'identification de la langue pour produire des systèmes robustes. Le fait que nous obtenions la première place pour les deux tâches auxquelles nous avons participé conforte les directions de recherche choisies.




(a) Tâche 1 (b) Tâche 3

FIGURE D.1 – Matrices de confusion du système soumis sur les ensembles d'évaluation (nombre d'enregistrements). Les détections multiples sont possibles.

# E

# Analyse des représentations de systèmes de reconnaissance de la parole

Nous avons vu que l'analyse d'un espace d'*embeddings* permettait de comprendre le fonctionnement des fonctions de régularisation utilisées pour les systèmes de reconnaissance de la langue. Dans le but de montrer que ce type d'analyse est largement généralisable, nous étudions dans cette annexe des espaces de représentation pour une autre tâche : la reconnaissance de la parole. Nous montrons que la mesure de divergences entre groupes d'*embeddings* dans un espace bien choisi peut rendre compte fidèlement de la performance attendue d'un système de reconnaissance de la parole sur un nouveau domaine, et ce de façon non supervisée. Ce travail a été effectué à la suite du *workshop SCALE 2020*, en collaboration avec Matthew Wiesner, John Steinberg et Kiran Karra.

## E.1 La tâche de reconnaissance de la parole

La tâche de reconnaissance de la parole consiste à produire une transcription écrite du signal d'entrée [Yu et Deng, 2015]. La mesure de performance est effectuée par une comparaison de la transcription produite avec le texte de référence. Pour les langues basées sur un alphabet, on utilise le *Word Error Rate* (*WER* - taux d'erreur mot). Il s'agit d'une distance d'édition entre les deux textes.

Nous nous intéressons ici à des systèmes de reconnaissance de la parole reposant sur un réseau de neurones de bout en bout (*end-to-end* en anglais) [Synnaeve *et al.*, 2020]. Un extracteur de *features* produit des représentations du signal audio d'entrée à l'échelle d'une trame. Un réseau de neurones profond est directement entraîné à prédire des séquences de sortie. Il utilise pour cela un ensemble d'étiquettes, pouvant être un alphabet, un ensemble de caractères, ou même des morceaux de phrases. Un modèle de langage, entraîné sur du texte, modélise l'enchaînement des mots. Un décodeur combine les informations apportées par le réseau de neurones et le modèle de langage pour proposer

les meilleures transcriptions du signal d'entrée.

## E.2 Motivation de l'analyse

Le réseau de neurones central des systèmes de reconnaissance de la parole à l'état de l'art est entraîné de façon supervisée [Gulati *et al.*, 2020]. Même si des résultats impressionnants ont été obtenus récemment avec un pré-apprentissage non supervisé de représentations [Baevski *et al.*, 2020], il n'en reste pas moins qu'une étape finale d'apprentissage supervisé est nécessaire et que la qualité des représentations dépend grandement du corpus d'apprentissage (que ce soit pour la phase d'apprentissage non supervisé ou supervisé) [Hsu *et al.*, 2021]. Il est donc inévitable que les systèmes subissent une perte de performance lorsqu'ils sont confrontés à un changement de domaine par rapport au corpus d'apprentissage [Seltzer *et al.*, 2013, Likhomanenko *et al.*, 2021].

Il est donc d'un grand intérêt pratique de prédire la performance d'un système sur un nouveau domaine, et ce de façon non supervisée. Supposons disposer d'un système de reconnaissance de la parole et vouloir l'utiliser dans un contexte applicatif particulier pour lequel nous sommes capables de collecter des données non annotées. Avec quel degré de confiance pouvons-nous appliquer notre système sur ce domaine ? Une réponse franche peut être obtenue en annotant une partie des données afin d'évaluer la performance. Nous proposons d'estimer cette performance de façon non supervisée en mesurant des divergences dans un espace d'*embeddings*.

Cette idée a été appliquée avec succès en traitement du texte pour des tâches d'analyse de sentiment et d'étiquetage morpho-syntaxique en utilisant un discriminateur dans un espace d'*embeddings* pour mesurer la divergence [Elsahar et Gallé, 2019]. Les différents domaines correspondaient alors à différentes catégories de produits ou différents sites *web*. Dans notre travail présenté ci-dessous, nous continuons à utiliser nos trois mesures de divergence, distance entre les moyennes, *CORAL* et *MMD*.

## E.3 Condition expérimentale

Nous étudions des systèmes de reconnaissance de la parole pour la langue anglaise. Ils sont implémentés avec l'outil *Espresso* [Wang *et al.*, 2019] et décrits dans le tableau E.1.

Nous travaillons sur la tâche de reconnaissance de la parole en langue anglaise. Nous entraînons des systèmes pour trois corpus d'entraînement : *librispeech* (100 heures), un sous-ensemble de *switchboard* (100 heures) et la concaténation de ces des deux corpus (200 heures). Un quatrième système est entraîné avec un apprentissage antagoniste dans le but de minimiser l'écart entre les deux domaines dans le corpus d'entraînement constitué de la concaténation de *librispeech* et *switchboard* [Ganin *et al.*, 2016].

Afin de rendre compte de la performance pour différents facteurs de variabilité, plusieurs corpus sont utilisés pour l'évaluation des systèmes. D'abord, nous évaluons la performance sur les ensembles de test associés à *librispeech* : *librispeech-test-clean*, *librispeech-test-other* et *librispeech-dev-other*. Ils permettent de rendre compte de la performance

TABLE E.1 – Architecture du système de reconnaissance de la parole de bout en bout.

| Réseau de neurones *de bout en bout* prenant en entrée des spectrogrammes en échelle *Mel* avec décodage *greedy* | | |
|---|---|---|
| Module | Modèle | Hyperparamètres |
| détection de parole | annotation manuelle | - |
| *features* acoustiques | spectrogrammes en échelle *Mel* | $\Delta_F = 25$ ms, $\delta_F = 10$ ms, 64 filtres |
| augmentations de données | aucune | - |
| réseau de neurones *end-to-end* | *Conformer* [Gulati *et al.*, 2020] | entraîné avec la *CTC loss* [Graves *et al.*, 2006] avec un alphabet de sortie défini par un *sentence piece model* [Kudo et Richardson, 2018] |
| modèle de langage | non utilisé | - |
| décodeur | glouton (*greedy*) | - |

pour de la parole lue, enregistrée sur microphone, avec des qualités de signal variables. Ensuite nous utilisons les corpus de test *switchboard* et *callhome* correspondant à des canaux téléphoniques. Finalement, nous tentons de rendre compte de deux autres facteurs de variabilité : le genre et l'accent. Pour cela nous utilisons le corpus *CommonVoice* dont nous extrayons six ensembles de test disjoints en fonction du genre (femmes et hommes) et de l'accent (américain, britannique ou indien) des locuteurs.

## E.4 Méthode d'analyse

Nous souhaitons mettre en évidence une corrélation entre des mesures de divergence, calculées de façon non supervisée dans un espace d'*embeddings*, et la performance en termes de reconnaissance de la parole.

Contrairement à la reconnaissance de la langue, la reconnaissance de la parole est une tâche dite *de séquence à séquence*. Le modèle ne produit donc pas de représentation de l'ensemble du signal d'entrée mais se limite à des représentations trame à trame, les séquences n'étant pas synchronisées. Nous utilisons deux espaces d'*embeddings* : la sortie du modèle *Conformer* ainsi qu'une couche intermédiaire, la sortie de l'encodeur. Pour chaque signal d'entrée et pour chacun de ces deux espaces, nous disposons donc d'une séquence de vecteurs.

D'autre part, comment regrouper ces vecteurs pour réaliser des ensembles cohérents afin de calculer des divergences ? Pour la reconnaissance de la langue, nous avions regroupé les *x-vectors* par classe de langue. En reconnaissance de la parole, des visualisations ont été réalisées dans la littérature afin de montrer des différences de réalisation d'un même phonème entre deux langues [Pratap *et al.*, 2020]. Une telle approche est comparable au travail que nous avons présenté dans le chapitre 5.

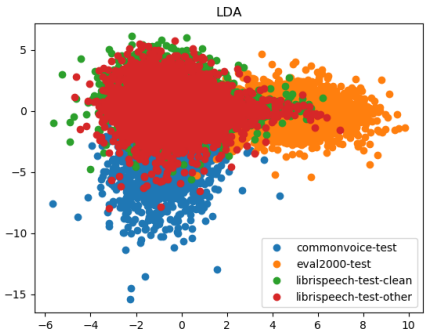



FIGURE E.1 – Projection *LDA* de l'espace d'*embeddings* du modèle entraîné sur *librispeech*. Cet espace correspond à la couche de sortie de l'encodeur du modèle *Conformer*.

Nous souhaitons cependant prédire la performance sur un nouveau domaine et donc disposer d'une métrique non supervisée. Nous ne pouvons donc pas utiliser les transcriptions pour assigner chaque trame à une étiquette phonétique. Cette assignation pourrait être réalisée en utilisant les transcriptions produites par le système de reconnaissance de la parole étudié. Nous décidons de ne pas emprunter cette voie car cela signifierait utiliser les prédictions d'un modèle dans le but d'étudier ses propres erreurs. Même si une telle approche bien maîtrisée peut fonctionner en pratique, nous préférons éviter cette source de confusion lors du développement d'une méthode d'analyse du système.

Nous utilisons donc la séquence des représentations trame à trame du signal sans distinguer de classes parmi ces vecteurs. Comment regrouper ces vecteurs afin de former des groupes d'*embeddings* correspondant à un domaine homogène permettant de calculer des divergences ? Trois options semblent possibles : regrouper uniquement les trames correspondant à un même enregistrement audio, regrouper les vecteurs provenant d'enregistrements d'un même locuteur, ou bien regrouper toutes les trames de tous les enregistrements d'un même corpus de test. Le regroupement par enregistrement est plus fin. Il ne permet cependant que de disposer d'une seule phrase pour l'estimation de la performance et d'un nombre faible de trames pour le calcul de la divergence. Dans la suite, nous nous limitons à des regroupements par corpus et par locuteur. Le regroupement par corpus semble suffisant pour voir apparaître un décalage entre les domaines. Nous l'observons pour une projection de quelques enregistrements de test pour le modèle entraîné sur *librispeech* avec une *LDA* réalisée sur la couche intermédiaire en figure E.1.

Enfin une divergence est mesurée entre deux groupes d'*embeddings*. Pour un groupe d'*embeddings* de test donné, avec quel autre groupe d'*embeddings* estimer les divergences dans le but d'obtenir une corrélation avec la performance ? Pour chaque système, nous utilisons comme ensemble de référence les *embeddings* associés à un corpus de validation de mêmes caractéristiques que le corpus d'entraînement du système, ce qui est supposé correspondre aux conditions idéales d'utilisation du système. Dans la suite, nous mesurons donc des divergences entre un nouveau domaine et le domaine correspondant aux conditions d'entraînement pour des espaces de représentations associés au modèle

*Conformer*. Nous souhaitons mettre en relation ces divergences avec la performance.

## E.5 Mesures de divergence par corpus

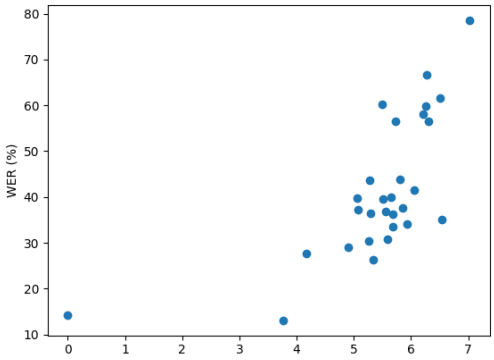
Dans un premier temps, nous calculons nos trois divergences entre ensembles d'*embeddings* correspondant aux différents corpus de test. La relation entre le *WER* et ces divergences mesurées par corpus est présentée en figure E.2 pour les deux espaces d'*embeddings* et les trois mesures de divergence. Le modèle est entraîné sur *librispeech* et l'ensemble de référence correspond au même corpus.

Sur la couche intermédiaire comme sur la couche de sortie, une corrélation semble apparaître entre la performance et la divergence avec l'ensemble de validation. Cette corrélation semble plus claire sur la couche de sortie et avec la distance entre les moyennes (figure E.2b) et la *MMD* (figure E.2f). Il apparaît donc que, comme pour la reconnaissance de la langue, une partie de la perte de performance des systèmes de reconnaissance de la parole se matérialise par un décalage dans les espaces de représentation.

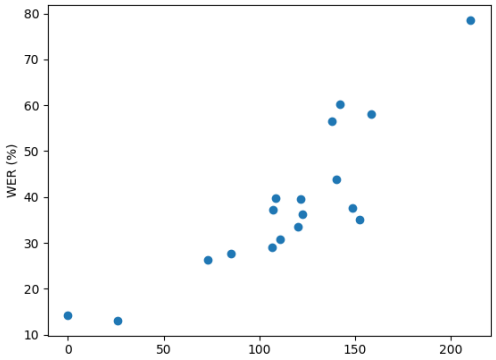
## E.6 Mesures de divergence par locuteur

Pouvons-nous aller plus loin que ce constat et, étant donné un corpus de test non annoté, estimer la performance du système sur ce corpus sur la base de mesures de divergence dans un espace d'*embeddings* ? Dans ce but, il est souhaitable de disposer de plus de paires ($divergence$,$WER$) afin de pouvoir entraîner un modèle prédisant le *WER* en fonction de la divergence. C'est pourquoi nous mesurons les divergences par groupe d'*embeddings* associés à un locuteur. Dans le but d'obtenir une estimation fiable du *WER* et de la divergence, nous nous limitons aux locuteurs pour lesquels nous disposons d'un nombre minimum d'enregistrements (vingt pour les expérimentations présentées en figure E.3). Remarquons que le regroupement par locuteur est effectué à des fins d'analyse. Dans un cas pratique où nous disposerions uniquement de données non annotées, il semble irréaliste de disposer d'annotations en locuteur.

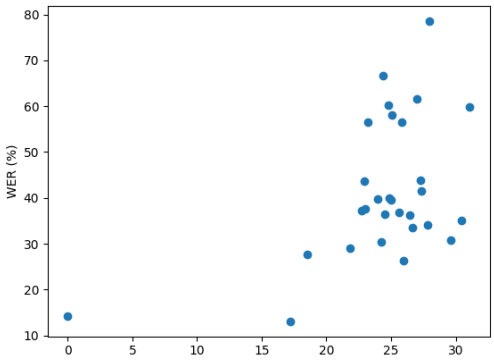
Pour le même modèle qu'en figure E.2, entraîné sur le corpus *librispeech*, nous présentons en figure E.3 les divergences avec l'ensemble de validation et *WER* associés, agrégés par locuteur. Chaque point représente donc un locuteur et les couleurs correspondent aux corpus de test. Nous entraînons un modèle simple de prédiction du *WER* à partir de la mesure de divergence. Il s'agit d'une régression linéaire, qui peut être évaluée en termes de coefficient de détermination $R^2$ [Barrett, 1974]. Une telle régression est estimée pour les trois mesures de divergence distance entre les moyennes, *CORAL* et *MMD*, et sur les deux couches du modèle déjà utilisées, une couche cachée et la couche de sortie. La meilleure prédiction par un modèle linéaire est ici obtenue pour la distance entre les moyennes sur la couche de sortie du modèle *Conformer* (figure E.3b).



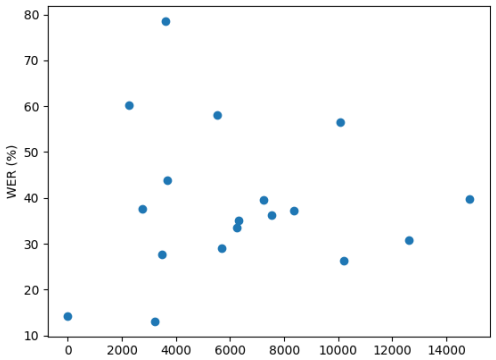

(a) Différence entre les moyennes, sortie de l'encodeur



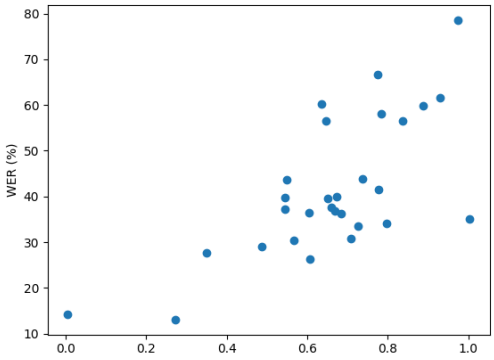

(b) Différence entre les moyennes, sortie du modèle *Conformer*



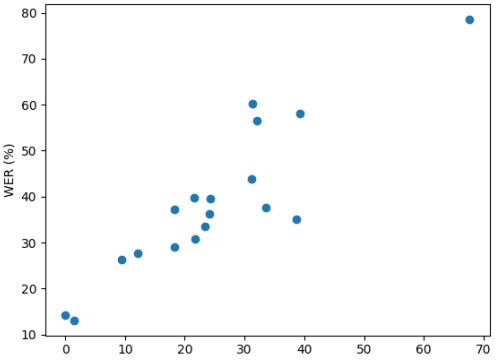

(c) *CORAL*, sortie de l'encodeur




(d) *CORAL*, sortie du modèle *Conformer*

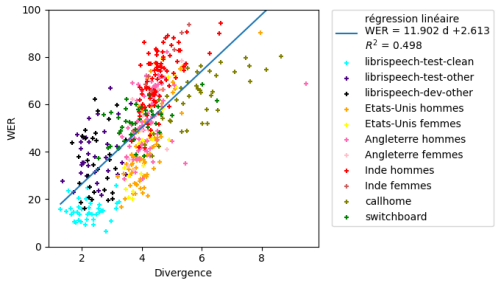



(e) *MMD*, sortie de l'encodeur

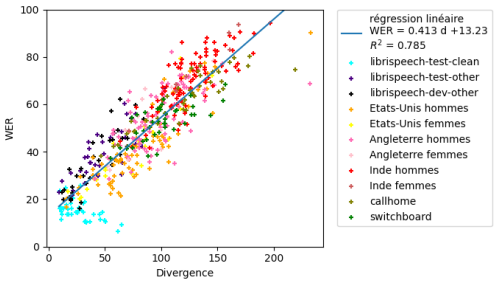



(f) *MMD*, sortie du modèle *Conformer*

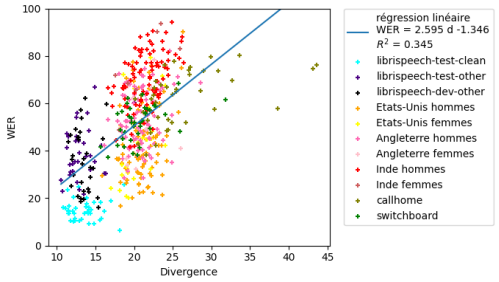
FIGURE E.2 – Divergences entre les *embeddings* correspondant au corpus de test et ceux du corpus de validation pour le modèle entraîné sur *librispeech*.



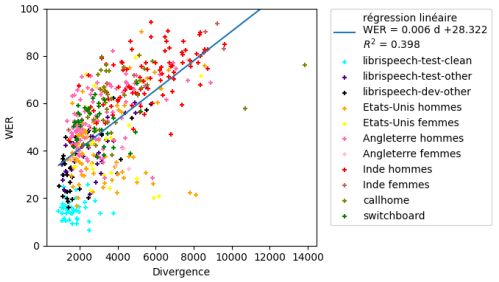

(a) Différence entre les moyennes, sortie de l'encodeur



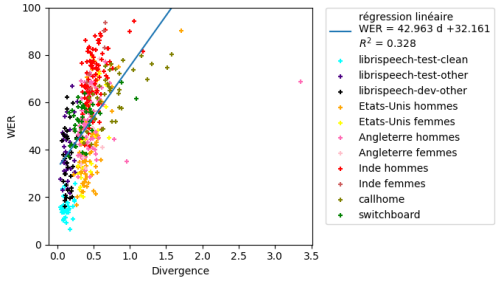

(b) Différence entre les moyennes, sortie du modèle *Conformer*



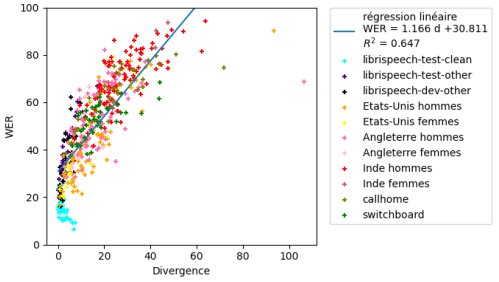

(c) *CORAL*, sortie de l'encodeur




(d) *CORAL*, sortie du modèle *Conformer*

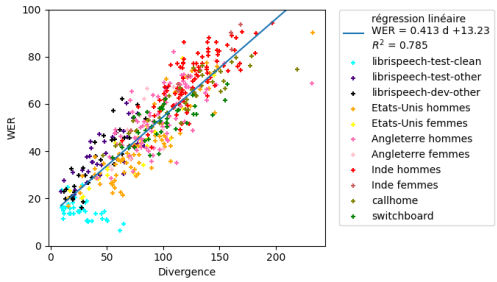



(e) *MMD*, sortie de l'encodeur

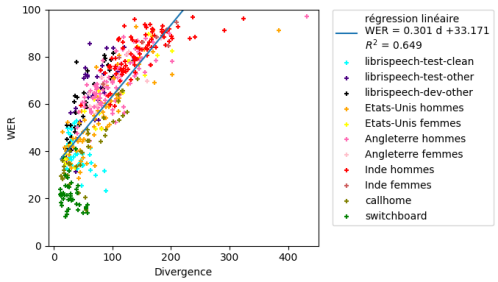



(f) *MMD*, sortie du modèle *Conformer*

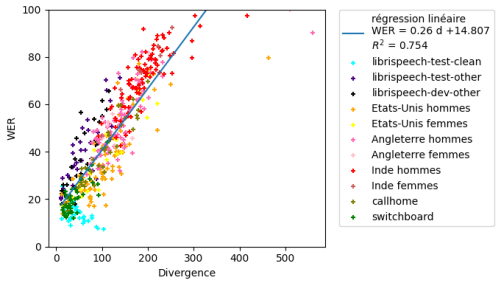
FIGURE E.3 – Divergences entre groupes d'*embeddings* pour le modèle entraîné sur le corpus *librispeech* pour différentes couches et mesures de divergence. Le corpus de référence pour le calcul des divergences est l'ensemble de validation de *librispeech*. Les *embeddings* sont regroupés par locuteur.



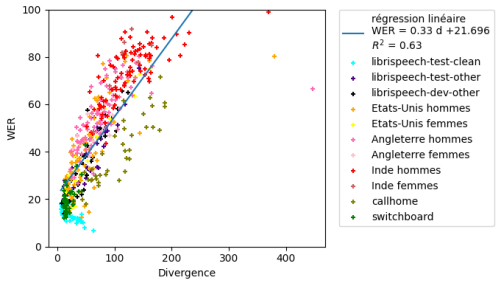

(a) Modèle entraîné sur *librispeech*.




(b) Modèle entraîné sur *switchboard*.




(c) Modèle entraîné sur *librispeech* et *switchboard*.




(d) Modèle entraîné sur *librispeech* et *switchboard* avec apprentissage antagoniste entre les domaines.

FIGURE E.4 – Divergences entre groupes d'*embeddings* pour la couche de sortie du modèle *Conformer*. La divergence utilisée est la distance entre les moyennes. Les *embeddings* sont regroupés par locuteur.

## E.7 Comparaison de systèmes

Pour cette mesure de divergence (distance entre les moyennes), nous observons des corrélations importantes entre divergence avec l'ensemble de validation sur la couche de sortie et performance. Nous l'avons observé pour quatre modèles entraînés sur les corpus *librispeech* (figure E.4a), *switchboard* (figure E.4b), sur un mélange des deux (figure E.4c) et sur ce même mélange des deux corpus mais avec un apprentissage antagoniste entre les deux corpus dans le but d'améliorer la robustesse du système (figure E.4d). Les corpus de test correspondant aux conditions de l'ensemble d'entraînement obtiennent toujours la meilleure performance. Les performances sur les autres corpus sont très fortement corrélées avec la divergence entre ces *embeddings* et l'ensemble de validation. Cependant cette corrélation n'est pas suffisante pour qu'un modèle linéaire permette de prédire de façon fiable la performance sur un nouveau domaine.

## E.8 Conclusion sur l'analyse d'un système de reconnaissance de la parole

Pour quatre modèles entraînés sur trois corpus, nous avons pu observer une corrélation entre des mesures de divergence dans des espaces de représentation du modèle *Conformer* et la performance finale de reconnaissance de la parole sur les modèles testés. La corrélation la plus claire a été obtenue en regroupant les *embeddings* par locuteur, en les observant sur la couche de sortie du modèle *Conformer* et en mesurant les divergences avec la distance entre les moyennes. Ces observations montrent que la méthode d'analyse des espaces de représentations intermédiaires de réseaux de neurones afin de rendre compte de variabilité, développée pour la reconnaissance de la langue, peut s'appliquer à d'autres tâches.

Les observations réalisées à ce jour ne permettent pas de réaliser une estimation fiable de la perte de performance sur un nouveau domaine à partir de mesures de divergence. Deux pistes de travail nous semblent intéressantes. La première est le développement d'un tel modèle de prédiction, en recourant à un modèle plus sophistiqué qu'une régression linéaire et éventuellement en utilisant plus de corpus de test. La seconde est d'utiliser des mesures non supervisées dans un espace d'*embeddings* dans le but de sélectionner des données pertinentes à incorporer à l'ensemble d'entraînement du modèle, dans une démarche d'apprentissage actif [Cohn *et al.*, 1996].

# Glossaire

**Activations** : sorties d'une couche intermédiaire d'un réseau de neurones, dépendant du signal d'entrée.

***Additive angular margin (AAM) softmax*** : fonction de coût de classification, raffinement de l'entropie croisée prenant en compte l'écart angulaire entre la représentation d'un échantillon et celle de la classe de référence.

**Adaptation de domaine** : scénario d'apprentissage faisant intervenir un domaine source et un domaine cible, correspondant aux conditions de l'ensemble de test, l'ensemble d'entraînement correspond majoritairement au domaine source, un ensemble limité ou non annoté du domaine cible est également disponible lors de l'apprentissage pour adapter le modèle.

**Adaptation de domaine non supervisée** : scénario d'adaptation de domaine pour lequel les données du domaine cible disponibles dans l'ensemble d'entraînement ne sont pas annotées.

**Adaptation de domaine supervisée** : scénario d'adaptation de domaine pour lequel les données du domaine cible disponibles dans l'ensemble d'entraînement sont annotées.

**Adaptation *feature based*** : méthode d'adaptation de domaine opérant sur les représentations du signal, en modifiant les représentations des données d'entraînement ou de test dans le but de faire coïncider les deux domaines.

**Adaptation *model based*** : méthode d'adaptation de domaine opérant sur les paramètres des modèles.

**Apprentissage multi-domaines** : scénario d'apprentissage faisant intervenir plusieurs domaines dans l'ensemble d'entraînement.

**Analyse en composantes principales, ACP** : méthode de projection non supervisée, consistant à sélectionner les directions expliquant la plus grande proportion de la variance d'une distribution.

**Analyse linéaire discriminante, *LDA*** : méthode de projection supervisée, consistant à sélectionner les directions les plus discriminantes pour un ensemble de données annotées en classes.

**Augmentation de données** : transformation des données d'entraînement dans le but d'augmenter leur diversité et par conséquent la robustesse du modèle.

***Bottleneck features*** : représentation du signal à l'échelle de la trame correspondant aux activations d'une couche d'un réseau de neurones entraîné pour une tâche

de reconnaissance de la parole.

**Calibration** : transformation des scores dans le but de leur donner une interprétation probabiliste, en reconnaissance de la langue, on produit souvent des rapports de vraisemblance.

**Classifieur final** : module prenant en entrée un *embedding* et produisant des scores.

**Corpus** : ensemble de données utilisé pour l'entraînement ou l'évaluation d'un système.

**Couche** : fonction paramétrique élémentaire (ensemble de cellules ou neurones), utilisée lors de la constitution de réseaux de neurones.

**Couche d'agrégation temporelle** : couche permettant de passer d'une séquence de trames portant une dimension temporelle à une représentation de taille fixe du signal, indépendamment de sa durée.

**Couche convolutive** : couche appliquant un ensemble de filtres locaux sur le signal, si les filtres sont appliqués uniquement selon la dimension temporelle, on parle de convolutions à une dimension, s'ils sont appliqués selon les dimensions temporelles et fréquentielles, on parle de convolutions à deux dimensions.

**Couche d'auto-attention** : couche prédisant un poids pour chaque dimension du signal d'entrée dans l'objectif de sélectionner certains motifs.

**Couche de *pooling*** : couche de réduction de la dimension, agrégeant certaines caractéristiques de la couche précédente.

**Couche pleine** : couche correspondant à une application affine avec une matrice pleine.

**Couche récurrente** : couche traitant séquentiellement les trames d'une séquence d'entrée.

***Correlation alignement (CORAL)*** : nom d'un ensemble de méthodes d'adaptation reposant sur la différence entre les matrices de covariance de deux domaines, il existe des méthodes *CORAL feature based* et *model based*.

**De bout en bout** : caractérise un système constitué d'un seul module.

**De séquence à séquence** : caractérise une tâche de prédiction d'une séquence à partir d'une séquence, comme la reconnaissance de la parole, la traduction ou la conversion de voix.

**Détection d'activité vocale** : segmentation d'un signal audio en segments de parole et d'absence de parole.

**Discriminant** : permettant de réaliser une séparation entre des classes.

**Divergence** : mesure de l'écart entre des distributions.

**Domaine** : distribution de signaux définie par certains facteurs de variabilité.

***Embedding*** : représentation de dimension fixe du signal à l'échelle du segment.

**Entraînement** : phase d'optimisation des paramètres d'un modèle.

**Entropie croisée** : une fonction de coût de classification.

**Époque** : passage sur l'ensemble du corpus d'entraînement au cours de la phase d'entraînement.

**Évaluation** : calcul de métriques caractérisant la performance d'un système, en général à l'aide d'un corpus d'évaluation.

**Facteur de variabilité** : variable affectant les caractéristiques du signal.

***Features*** : représentation vectorielle du signal, utile à une tâche visée, dans ce travail,

les *features* font toujours référence à une représentation à l'échelle de la trame.

***Features* acoustiques** : *features* calculés à partir de connaissances en traitement de signal, par exemple basés sur un spectrogramme.

**Fonction de coût** : critère calculé à partir des données d'entraînement et des paramètres d'un réseau de neurones, ce critère est minimisé au cours de l'apprentissage, la fonction de coût doit être différentiable et pouvoir être estimée à partir de *minibatchs*.

**Généralisation à un domaine inconnu** : scénario d'apprentissage pour lequel le système est testé sur un domaine non observé dans l'ensemble d'entraînement.

***Gated recurrent unit (GRU)*** : type de paramétrisation de couches récurrentes.

**Invariant** : caractérise des représentations pour lesquelles un facteur de variabilité est négligeable devant l'écart entre les classes.

***Long short-term memory (LSTM)*** : type de paramétrisation de couches récurrentes.

***Metric learning*** : famille de fonctions de coût basées sur les distances entre les représentations de paires de signaux.

**Métrique** : quantité utilisée pour évaluer la performance d'un système et évaluée sur un corpus d'évaluation.

***Minibatch*** : ensemble réduit des données d'entraînement, échantillonné à chaque étape de la descente de gradient stochastique lors de l'entraînement d'un réseau de neurones.

**Multi-tâches** : méthode d'entraînement combinant plusieurs tâches pour optimiser les paramètres d'un modèle.

***National Institue of Standards and Technology (NIST)*** : institution gouvernementale des États-Unis d'Amérique, organisant régulièrement depuis 1996 des campagnes internationales d'évaluation de reconnaissance de la langue.

***N-pair loss*** : fonction de coût de *metric learning* reposant sur l'échantillonnage de $n$ paires de signaux, où $n$ est le nombre de classes.

***Oriental Language Recognition (OLR)*** : série de campagnes internationales d'évaluation de reconnaissance de la langue organisée depuis 2016 par les institutions *Xiamen University*, *Tsinghua University*, *Duke-Kunshan University*, *Northwestern Polytechnical University* et *Speechocean*.

**Perceptron multi-couches** : réseau de neurones constitué d'une succession de couches pleines.

**Reconnaissance de la langue** : tâche consistant à prédire la langue utilisée dans un signal audio.

**Reconnaissance de la parole** : tâche consistant à produire une transcription d'un signal audio.

**Régularisation** : méthode appliquée à un modèle pour en réduire la capacité dans le but d'augmenter la généralisation.

**Représentation** : vecteur obtenu par une transformation du signal d'entrée, utilisé par le système pour réaliser la tâche.

**Réseau de neurones** : type de modèle paramétrique constitué d'un enchaînement de couches élémentaires, dont les paramètres peuvent être appris conjointement.

**Réseau de neurones récurrent, *RNN*** : architecture de réseau de neurones constituée d'un enchaînement de couches récurrentes.

**Réseau résiduel, *ResNet*** : architecture de réseau de neurones pour laquelle les sorties de certaines couches du réseau sont ajoutées aux sorties de couches plus profondes, dans le but de faciliter la rétro-propagation du gradient.

**Scénario d'apprentissage** : terme utilisé dans cette thèse pour faire référence aux configurations d'apprentissage définies par les domaines présents dans les ensembles d'entraînement et de test.

**Score** : scalaire produit par un système caractérisant la confiance dans une prédiction.

**Segment** : échelle de temps correspondant à l'ensemble du signal d'entrée.

***Stacked bottleneck features*** : type de *bottleneck features* obtenu par l'application successive de plusieurs réseaux de neurones.

***T-distributed stochastic neighbor embedding (t-SNE)*** : méthode de projection reposant sur le respect de la distribution des distances entre paires d'échantillons.

**Tâche** : nature des prédictions devant être réalisées par un système.

***Time delay neural network (TDNN)*** : architecture de réseau de neurones reposant sur des convolutions à une dimension avec dilatation

***Trame*** : échelle de temps de l'ordre de la dizaine de millisecondes.

***Transformer*** : architecture de réseau de neurones constituée d'une succession de couches d'auto-attention.

***Triplet loss*** : fonction de coût de *metric learning* basée sur l'échantillonnage de triplets correspondant à deux signaux de même classe et un signal d'une classe différente.

**Vérification du locuteur** : tâche de vérification consistant à déterminer si un signal de test a été prononcé par le même locuteur qu'un ensemble de signaux de référence.

***X-vector*** : *embedding* extrait d'un réseau de neurones entraîné pour une tâche d'identification de la langue.

# Publications

[Duroselle *et al.*, 2020a] Duroselle, R., Jouvet, D. et Illina, I. (2020a). Adaptation de domaine non supervisée pour la reconnaissance de la langue par régularisation d'un réseau de neurones. *In Actes des JEP 2020 - Journées d'Études sur la Parole*, pages 190–198. ATALA ; AFCP.

[Duroselle *et al.*, 2020b] Duroselle, R., Jouvet, D. et Illina, I. (2020b). Metric learning loss functions to reduce domain mismatch in the x-vector space for language recognition. *In Proceedings of Interspeech 2020 - Annual Conference of the International Speech Communication Association.*

[Duroselle *et al.*, 2020c] Duroselle, R., Jouvet, D. et Illina, I. (2020c). Unsupervised regularization of the embedding extractor for robust language identification. *In Proceedings of Odyssey 2020 - The Speaker and Language Recognition Workshop.*

[Duroselle *et al.*, 2021a] Duroselle, R., Sahidullah, M., Jouvet, D. et Illina, I. (2021a). Language recognition on unknown conditions : the LORIA-Inria-MULTISPEECH system for AP20-OLR Challenge. *In Proceedings of Interspeech 2021 - Annual Conference of the International Speech Communication Association.*

[Duroselle *et al.*, 2021b] Duroselle, R., Sahidullah, M., Jouvet, D. et Illina, I. (2021b). Modeling and training strategies for language recognition systems. *In Proceedings of Interspeech 2021 - Annual Conference of the International Speech Communication Association.*

# Bibliographie


[Abdullah *et al.*, 2020] Abdullah, B. M., Avgustinova, T., Möbius, B. et Klakow, D. (2020). Cross-domain adaptation of spoken language identification for related languages : the curious case of slavic languages. *In Proceedings of Interspeech 2020 - Annual Conference of the International Speech Communication Association*, pages 477–481.

[Achille et Soatto, 2018] Achille, A. et Soatto, S. (2018). Emergence of invariance and disentanglement in deep representations. *In Proceedings of ITA 2018 - Information Theory and Applications Workshop*. IEEE.

[Adams *et al.*, 2019] Adams, O., Wiesner, M., Watanabe, S. et Yarowsky, D. (2019). Massively multilingual adversarial speech recognition. *In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 96–108.

[Alam *et al.*, 2018] Alam, J., Bhattacharya, G. et Kenny, P. (2018). Speaker verification in mismatched conditions with frustratingly easy domain adaptation. *In Proceedings of Odyssey 2018 - The Speaker and Language Recognition Workshop*, pages 176–180.

[Amodei *et al.*, 2016] Amodei, D., Ananthanarayanan, S., Anubhai, R., Bai, J., Battenberg, E., Case, C., Casper, J., Catanzaro, B., Cheng, Q., Chen, G. *et al.* (2016). Deep speech 2 : end-to-end speech recognition in English and Mandarin. *In Proceedings of ICML 2016 - International Conference on machine learning*, pages 173–182. PMLR.

[Ardila *et al.*, 2020] Ardila, R., Branson, M., Davis, K., Kohler, M., Meyer, J., Henretty, M., Morais, R., Saunders, L., Tyers, F. et Weber, G. (2020). Common voice : a massively-multilingual speech corpus. *In Proceedings of LREC 2020 - Language Resources and Evaluation Conference*, pages 4218–4222.

[Arjovsky, 2020] Arjovsky, M. (2020). *Out of distribution generalization in machine learning*. Thèse de doctorat, New York University.

[Arjovsky *et al.*, 2020] Arjovsky, M., Bottou, L., Gulrajani, I. et Lopez-Paz, D. (2020). Invariant risk minimization. *In arXiv :1907.02893 [cs, stat]*.

[Baevski *et al.*, 2020] Baevski, A., Zhou, Y., Mohamed, A. et Auli, M. (2020). Wav2vec 2.0 : a framework for self-supervised learning of speech representations. *In Advances in Neural Information Processing Systems*, volume 33.

[Bahmaninezhad *et al.*, 2021] Bahmaninezhad, F., Zhang, C. et Hansen, J. H. L. (2021). An investigation of domain adaptation in speaker embedding space for speaker recognition. *In Speech Communication*, volume 129, pages 7–16.

[Barrett, 1974] Barrett, J. P. (1974). The coefficient of determination —some limitations. *In The American Statistician*, volume 28, pages 19–20. Taylor & Francis.

[Ben-David *et al.*, 2010] Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F. et Vaughan, J. W. (2010). A theory of learning from different domains. *In Machine learning*, volume 79, pages 151–175. Springer.

[Ben-David *et al.*, 2007] Ben-David, S., Blitzer, J., Crammer, K. et Pereira, F. (2007). Analysis of representations for domain adaptation. *In Advances in neural information processing systems*, pages 137–144.

[Ben-Reuven et Goldberger, 2016] Ben-Reuven, E. et Goldberger, J. (2016). A semisupervised approach for language identification based on ladder networks. *In Proceedings of Odyssey 2016 - The Speaker and Language Recognition Workshop*, pages 319–325.

[BenZeghiba *et al.*, 2012] BenZeghiba, M. F., Gauvain, J.-L. et Lamel, L. (2012). Phonotactic language recognition using MLP features. *In Proceedings of Interspeech 2012 - Annual Conference of the International Speech Communication Association.*

[Bhattacharya *et al.*, 2019a] Bhattacharya, G., Alam, J. et Kenny, P. (2019a). Adapting end-to-end neural speaker verification to new languages and recording conditions with adversarial training. *In Proceedings of ICASSP 2019 - IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6041–6045. IEEE.

[Bhattacharya *et al.*, 2019b] Bhattacharya, G., Monteiro, J., Alam, J. et Kenny, P. (2019b). Generative adversarial speaker embedding networks for domain robust end-to-end speaker verification. *In Proceedings of ICASSP 2019 - IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6226–6230.

[Bottou, 2012] Bottou, L. (2012). Stochastic gradient descent tricks. *In Neural Networks : Tricks of the Trade*, pages 421–436. Springer Berlin Heidelberg.

[Bousquet et Rouvier, 2019] Bousquet, P.-M. et Rouvier, M. (2019). On robustness of unsupervised domain adaptation for speaker recognition. *In Proceedings of Interspeech 2019 - Annual Conference of the International Speech Communication Association*, pages 2958–2962.

[Bousquet et Rouvier, 2020] Bousquet, P.-M. et Rouvier, M. (2020). Adaptation strategy and clustering from scratch for new domains of speaker recognition. *In Proceedings of Odyssey 2020 - The Speaker and Language Recognition Workshop*, pages 81–87. ISCA.

[Brümmer, 2007] Brümmer, N. (2007). The FoCal toolkit. *In https://sites.google.com/site/nikobrummer/focalmulticlass*.

[Brümmer, 2010] Brümmer, N. (2010). *Measuring, refining and calibrating speaker and language information extracted from speech.* Thèse de doctorat, University of Stellenbosch.

[Brümmer et Van Leeuwen, 2006] Brümmer, N. et Van Leeuwen, D. A. (2006). On calibration of language recognition scores. *In Proceedings of Odyssey 2006 - The Speaker and Language Recognition Workshop*, pages 1–8.

[Cai *et al.*, 2020a] Cai, D., Cai, W. et Li, M. (2020a). Within-sample variability-invariant loss for robust speaker recognition under noisy environments. *In Proceedings of ICASSP 2020 - IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6469–6473. IEEE.

[Cai *et al.*, 2019] Cai, W., Cai, D., Huang, S. et Li, M. (2019). Utterance-level end-to-end language identification using attention-based CNN-BLSTM. *In Proceedings of ICASSP 2019 - IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5991–5995.

[Cai *et al.*, 2018a] Cai, W., Cai, Z., Liu, W., Wang, X. et Li, M. (2018a). Insights in-to-end learning scheme for language identification. *In Proceedings of ICASSP 2018 - IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5209–5213. IEEE.

[Cai *et al.*, 2018b] Cai, W., Cai, Z., Zhang, X., Wang, X. et Li, M. (2018b). A novel learnable dictionary encoding layer for end-to-end language identification. *In Proceedings of ICASSP 2018 - IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5189–5193, Calgary, AB. IEEE.

[Cai *et al.*, 2018c] Cai, W., Chen, J. et Li, M. (2018c). Exploring the encoding layer and loss function in end-to-end speaker and language recognition system. *In Proceedings of Odyssey 2018 - The Speaker and Language Recognition Workshop*, pages 74–81.

[Cai *et al.*, 2020b] Cai, W., Chen, J., Zhang, J. et Li, M. (2020b). On-the-fly data loader and utterance-level aggregation for speaker and language recognition. *In IEEE/ACM Transactions on Audio, Speech, and Language Processing*, volume 28, pages 1038–1051. IEEE.

[Campbell *et al.*, 2006] Campbell, W., Sturim, D. et Reynolds, D. (2006). Support vector machines using GMM supervectors for speaker verification. *In IEEE Signal Processing Letters*, volume 13, pages 308–311.

[Campbell *et al.*, 2008] Campbell, W. M., Sturim, D. E., Torres-Carrasquillo, P. A. et Reynolds, D. A. (2008). A comparison of subspace feature-domain methods for language recognition. *In Proceedings of Interspeech 2008 - Annual Conference of the International Speech Communication Association.*

[Chen *et al.*, 2018] Chen, Z., Badrinarayanan, V., Lee, C.-Y. et Rabinovich, A. (2018). Gradnorm : gradient normalization for adaptive loss balancing in deep multi-task networks. *In Proceedings of ICML 2018 - International Conference on Machine Learning*, pages 794–803. PMLR.

[Chettri *et al.*, 2021] Chettri, B., Hautamäki, R., Sahidullah, M. et Kinnunen, T. (2021). Data quality as predictor of voice anti-spoofing generalization. *In Proceedings of Interspeech 2021 - Annual Conference of the International Speech Communication Association.*

[Choi et Wang, 2021] Choi, K. et Wang, Y. (2021). Listen, read, and identify : multimodal singing language identification of music. *In arXiv :2103.01893 [cs, eess]*.

[Chowdhury *et al.*, 2020] Chowdhury, S. A., Ali, A., Shon, S. et Glass, J. (2020). What does an end-to-end dialect identification model learn about non-dialectal information ? *In Proceedings of Interspeech 2020 - Annual Conference of the International Speech Communication Association*, pages 462–466. ISCA.

[Chung *et al.*, 2020] Chung, J. S., Huh, J., Mun, S., Lee, M., Heo, H.-S., Choe, S., Ham, C., Jung, S., Lee, B.-J. et Han, I. (2020). In defence of metric learning for speaker recognition. *In Proceedings of Interspeech 2020 - Annual Conference of the International Speech Communication Association*, pages 2977–2981.

[Cohn *et al.*, 1996] Cohn, D. A., Ghahramani, Z. et Jordan, M. I. (1996). Active learning with statistical models. *In Journal of artificial intelligence research*, volume 4, pages 129–145.

[Damodaran *et al.*, 2018] Damodaran, B. B., Kellenberger, B., Flamary, R., Tuia, D. et Courty, N. (2018). DeepJDOT : deep joint distribution optimal transport for unsupervised domain adaptation. *In Proceedings of ECCV 2018 - European Conference on Computer Vision*, pages 467–483. Cham.

[Davis et Mermelstein, 1980] Davis, S. et Mermelstein, P. (1980). Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *In IEEE Transactions on Acoustics, Speech, and Signal Processing*, volume 28, pages 357–366.

[Dehak *et al.*, 2011] Dehak, N., Torres-Carrasquillo, P. A., Reynolds, D. et Dehak, R. (2011). Language recognition via i-vectors and dimensionality reduction. *In Proceedings of Interspeech 2011 - Annual Conference of the International Speech Communication Association*.

[Deng *et al.*, 2019] Deng, J., Guo, J., Xue, N. et Zafeiriou, S. (2019). Arcface : additive angular margin loss for deep face recognition. *In Proceedings of CVPR 2019 - Conference on Computer Vision and Pattern Recognition*, pages 4690–4699.

[Desplanques *et al.*, 2020] Desplanques, B., Thienpondt, J. et Demuynck, K. (2020). ECAPA-TDNN : emphasized channel attention, propagation and aggregation in TDNN based speaker verification. *In Proceedings of Interspeech 2020 - Annual Conference of the International Speech Communication Association*, pages 3830–3834.

[Elsahar et Gallé, 2019] Elsahar, H. et Gallé, M. (2019). To annotate or not ? *In Proceedings of EMNLP-IJCNLP 2019 - Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 2163–2173, Hong Kong, China. Association for Computational Linguistics.

[Eskimez *et al.*, 2018] Eskimez, S. E., Soufleris, P., Duan, Z. et Heinzelman, W. (2018). Front-end speech enhancement for commercial speaker verification systems. *In Speech Communication*, volume 99, pages 101–113.

[Fér *et al.*, 2017] Fér, R., Matějka, P., Grézl, F., Plchot, O., Veselý, K. et Černocký, J. H. (2017). Multilingually trained bottleneck features in spoken language recognition. *In Computer Speech & Language*, volume 46, pages 252–267. Elsevier.

[Fernando *et al.*, 2018] Fernando, S., Sethu, V. et Ambikairajah, E. (2018). Sub-band envelope features using frequency domain linear prediction for short duration language identification. *In Proceedings of Interspeech 2018 - Annual Conference of the International Speech Communication Association*, pages 1818–1822. ISCA.

[Feydy *et al.*, 2019] Feydy, J., Séjourné, T., Vialard, F.-X., Amari, S.-i., Trouvé, A. et Peyré, G. (2019). Interpolating between optimal transport and MMD using Sinkhorn divergences. *In Proceedings of AISTATS 2019 - International Conference on Artificial Intelligence and Statistics*, pages 2681–2690.

[Fiscus *et al.*, 2000] Fiscus, J. G., Fisher, W. M., Martin, A. F., Przybocki, M. A. et Pallett, D. S. (2000). 2000 NIST evaluation of conversational speech recognition over the telephone : English and mandarin performance results. *In Proceedings of Speech Transcription Workshop 2000*.

[Frederiksen *et al.*, 2018] Frederiksen, P. S., Villalba, J., Watanabe, S., Tan, Z.-H. et Dehak, N. (2018). Effectiveness of single-channel BLSTM enhancement for language identification. *In Proceedings of Interspeech 2018 - Annual Conference of the International Speech Communication Association*, pages 1823–1827. ISCA.

[Ganin *et al.*, 2016] Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M. et Lempitsky, V. (2016). Domain-adversarial training of neural networks. *In The Journal of Machine Learning Research*, volume 17, pages 2096–2030.

[Garcia-Romero et Espy-Wilson, 2011] Garcia-Romero, D. et Espy-Wilson, C. Y. (2011). Analysis of i-vector length normalization in speaker recognition systems. *In Proceedings of Interspeech 2011 - Annual Conference of the International Speech Communication Association*.

[Garcia-Romero *et al.*, 2020] Garcia-Romero, D., Sell, G. et Mccree, A. (2020). MagNetO : x-vector magnitude estimation network plus offset for improved speaker recognition. *In Proceedings of Odyssey 2020 - The Speaker and Language Recognition Workshop*. ISCA.

[Gelly, 2017] Gelly, G. (2017). *Réseaux de neurones récurrents pour le traitement automatique de la parole*. Thèse de doctorat, Université Paris-Sud.

[Gelly et Gauvain, 2017] Gelly, G. et Gauvain, J.-L. (2017). Spoken language identification using LSTM-based angular proximity. *In Proceedings of Interspeech 2017 - Annual Conference of the International Speech Communication Association*, pages 2566–2570.

[Gelly *et al.*, 2016] Gelly, G., Gauvain, J.-L., Le, V. et Messaoudi, A. (2016). A divide-and-conquer approach for language identification based on recurrent neural networks. *In Proceedings of Interspeech 2016 - Annual Conference of the International Speech Communication Association*, pages 3231–3235.

[Geng *et al.*, 2016] Geng, W., Wang, W., Zhao, Y., Cai, X. et Xu, B. (2016). End-to-end language identification using attention-based recurrent neural networks. *In Proceedings of Interspeech 2016 - Annual Conference of the International Speech Communication Association*, pages 2944–2948.

[Glembek *et al.*, 2008] Glembek, O., Matějka, P., Burget, L. et Mikolov, T. (2008). Advances in phonotactic language recognition. *In Proceedings of Interspeech 2008 - Annual Conference of the International Speech Communication Association.*

[Godfrey *et al.*, 1992] Godfrey, J. J., Holliman, E. C. et McDaniel, J. (1992). Switchboard : telephone speech corpus for research and development. *In Proceedings of ICASSP 1992 - IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 517–520. IEEE Computer Society.

[Gonzalez-Dominguez *et al.*, 2015] Gonzalez-Dominguez, J., Lopez-Moreno, I., Moreno, P. J. et Gonzalez-Rodriguez, J. (2015). Frame-by-frame language identification in short utterances using deep neural networks. *In Neural Networks*, volume 64, pages 49–58.

[Goodfellow *et al.*, 2016] Goodfellow, I., Bengio, Y. et Courville, A. (2016). *Deep Learning.* MIT Press. http://www.deeplearningbook.org.

[Goodfellow *et al.*, 2020] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. et Bengio, Y. (2020). Generative adversarial networks. *In Communications of the ACM*, volume 63, pages 139–144. ACM New York, NY, USA.

[Graves *et al.*, 2006] Graves, A., Fernández, S., Gomez, F. et Schmidhuber, J. (2006). Connectionist temporal classification : labelling unsegmented sequence data with recurrent neural networks. *In Proceedings of ICML 2006 - International Conference on Machine Learning*, pages 369–376.

[Graves et Jaitly, 2014] Graves, A. et Jaitly, N. (2014). Towards end-to-end speech recognition with recurrent neural networks. *In Proceedings of ICML 2014 - International Conference on machine learning*, pages 1764–1772. PMLR.

[Greenberg *et al.*, 2020] Greenberg, C. S., Mason, L. P., Sadjadi, S. O. et Reynolds, D. A. (2020). Two decades of speaker recognition evaluation at the National Institute of Standards and Technology. *In Computer Speech & Language*, volume 60, page 101032. Elsevier.

[Gulati *et al.*, 2020] Gulati, A., Qin, J., Chiu, C.-C., Parmar, N., Zhang, Y., Yu, J., Han, W., Wang, S., Zhang, Z., Wu, Y. et Pang, R. (2020). Conformer : convolution-augmented transformer for speech recognition. *In Proceedings of Interspeech 2020 - Annual Conference of the International Speech Communication Association*, pages 5036–5040.

[Gulrajani et Lopez-Paz, 2020] Gulrajani, I. et Lopez-Paz, D. (2020). In search of lost domain generalization. *In Proceedings of ICLR 2020 - International Conference on Learning Representations.*

[Haeusser *et al.*, 2017] Haeusser, P., Frerix, T., Mordvintsev, A. et Cremers, D. (2017). Associative domain adaptation. *In Proceedings of ICCV 2017 - IEEE International Conference on Computer Vision*, pages 2784–2792, Venice. IEEE.

[Hammarström *et al.*, 2021] Hammarström, H., Forkel, R., Haspelmath, M. et Bank, S. (2021). Glottolog 4.4. *In https://glottolog.org*.

[Heigold *et al.*, 2016] Heigold, G., Moreno, I., Bengio, S. et Shazeer, N. (2016). End-to-end text-dependent speaker verification. *In Proceedings of ICASSP 2016 - IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5115–5119. IEEE.

[Hoffer et Ailon, 2015] Hoffer, E. et Ailon, N. (2015). Deep metric learning using triplet network. *In Proceedings of International Workshop on Similarity-Based Pattern Recognition*, pages 84–92. Springer.

[House et Neuburg, 1977] House, A. S. et Neuburg, E. P. (1977). Toward automatic identification of the language of an utterance. I. Preliminary methodological considerations. *In The Journal of the Acoustical Society of America*, volume 62, pages 708–713. Acoustical Society of America.

[Hsu *et al.*, 2021] Hsu, W.-N., Sriram, A., Baevski, A., Likhomanenko, T., Xu, Q., Pratap, V., Kahn, J., Lee, A., Collobert, R., Synnaeve, G. et Auli, M. (2021). Robust wav2vec 2.0 : analyzing domain shift in self-supervised pre-training. *In arXiv :2104.01027 [cs, eess]*.

[Izmailov *et al.*, 2018] Izmailov, P., Podoprikhin, D., Garipov, T., Vetrov, D. et Wilson, A. G. (2018). Averaging weights leads to wider optima and better generalization. *In Proceedings of UAI 2018 - Conference on Uncertainty in Artificial Intelligence*, pages 876–885.

[Jančík *et al.*, 2010] Jančík, Z., Plchot, O., Brümmer, N., Burget, L., Glembek, O., Hubeika, V., Karafiát, M., Matějka, P., Mikolov, T., Strasheim, A. et Černocký, J. H. (2010). Data selection and calibration issues in automatic language recognition-investigation with BUT-AGNITIO NIST LRE 2009 system. *In Proceedings of Odyssey 2010 - The Speaker and Language Recognition Workshop*, page 37.

[Jiang *et al.*, 2014] Jiang, B., Song, Y., Wei, S., Liu, J.-H., McLoughlin, I. V. et Dai, L.-R. (2014). Deep bottleneck features for spoken language identification. *In PloS one*, volume 9, page e100795. Public Library of Science.

[Joshi *et al.*, 2012] Joshi, M., Dredze, M., Cohen, W. et Rose, C. (2012). Multi-domain learning : when do domains matter ? *In Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1302–1312.

[Kadiri, 1998] Kadiri, N. (1998). *Conséquences d'un environnement bruité sur la production de la parole*. Thèse de doctorat, Université de Toulouse 3.

[Kenny, 2005] Kenny, P. (2005). Joint factor analysis of speaker and session variability : theory and algorithms. *In CRIM, Montreal, (Report) CRIM-06/08-13*, volume 14, page 2.

[Kingma et Ba, 2017] Kingma, D. P. et Ba, J. (2017). Adam : a method for stochastic optimization. *In arXiv :1412.6980 [cs]*.

[Ko *et al.*, 2017] Ko, T., Peddinti, V., Povey, D., Seltzer, M. L. et Khudanpur, S. (2017). A study on data augmentation of reverberant speech for robust speech recognition. *In Proceedings of ICASSP 2017 - IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5220–5224, New Orleans, LA. IEEE.

[Krizhevsky *et al.*, 2017] Krizhevsky, A., Sutskever, I. et Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. *In Communications of the ACM*, volume 60, pages 84–90.

[Kudo et Richardson, 2018] Kudo, T. et Richardson, J. (2018). SentencePiece : a simple and language independent subword tokenizer and detokenizer for neural text processing. *In Proceedings of EMNLP 2018 - Conference on Empirical Methods in Natural Language Processing*.

[Kulkarni *et al.*, 2020] Kulkarni, A., Colotte, V. et Jouvet, D. (2020). Transfer learning of the expressivity using FLOW metric learning in multispeaker text-to-speech synthesis. *In Proceedings of Interspeech 2020 - Annual Conference of the International Speech Communication Association*.

[Kull et Flach, 2014] Kull, M. et Flach, P. (2014). Patterns of dataset shift. *In First International Workshop on Learning over Multiple Contexts (LMCE) at ECML-PKDD*.

[Lee *et al.*, 2019] Lee, K. A., Wang, Q. et Koshinaka, T. (2019). The CORAL+ algorithm for unsupervised domain adaptation of PLDA. *In Proceedings of ICASSP 2019 - IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5821–5825.

[Li *et al.*, 2013] Li, H., Ma, B. et Lee, K. A. (2013). Spoken language recognition : from fundamentals to practice. *In Proceedings of the IEEE*, volume 101, pages 1136–1159. IEEE.

[Li *et al.*, 2021] Li, L., Li, Z., Liu, Y. et Hong, Q. (2021). Deep joint learning for language recognition. *In Neural Networks*. Elsevier.

[Li *et al.*, 2020a] Li, Z., Zhao, M., Hong, Q., Li, L., Tang, Z., Wang, D., Song, L. et Yang, C. (2020a). AP20-OLR challenge : three tasks and their baselines. *In Proceedings of 2020 APSIPA ASC - Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, pages 550–555. IEEE.

[Li *et al.*, 2020b] Li, Z., Zhao, M., Li, J., Li, L. et Hong, Q. (2020b). On the usage of multi-feature integration for speaker verification and language identification. *In Proceedings of Interspeech 2020 - Annual Conference of the International Speech Communication Association*, pages 457–461. ISCA.

[Li *et al.*, 2020c] Li, Z., Zhao, M., Li, J., Zhi, Y., Li, L. et Hong, Q. (2020c). The XMUSPEECH system for the AP19-OLR challenge. *In Proceedings of Interspeech 2020 - Annual Conference of the International Speech Communication Association*, pages 452–456. ISCA.

[Likhomanenko *et al.*, 2021] Likhomanenko, T., Xu, Q., Pratap, V., Tomasello, P., Kahn, J., Avidov, G., Collobert, R. et Synnaeve, G. (2021). Rethinking evaluation in ASR : are our models robust enough ? *In arXiv :2010.11745 [cs, eess]*.

[Lin *et al.*, 2018] Lin, W.-W., Mak, M.-W., Li, L. et Chien, J.-T. (2018). Reducing domain mismatch by maximum mean discrepancy based autoencoders. *In Proceedings of Odyssey 2018 - The Speaker and Language Recognition Workshop*, pages 162–167.

[Ling *et al.*, 2020] Ling, S., Salazar, J., Liu, Y., Kirchhoff, K. et Amazon, A. (2020). Bertphone : Phonetically-aware encoder representations for utterance-level speaker and language recognition. *In Proceedings of Odyssey 2020 - The Speaker and Language Recognition Workshop*, pages 9–16.

[Long *et al.*, 2015] Long, M., Cao, Y., Wang, J. et Jordan, M. I. (2015). Learning transferable features with deep adaptation networks. *In Proceedings of ICML 2015 - International Conference on Machine Learning*, pages 97–105.

[Lopez *et al.*, 2018] Lopez, J. A. V., Brummer, N. et Dehak, N. (2018). End-to-end versus embedding neural networks for language recognition in mismatched conditions. *In Proceedings of Odyssey 2018 - The Speaker and Language Recognition Workshop*, pages 112–119. ISCA.

[Lozano-Diez, 2018] Lozano-Diez, A. (2018). *Bottleneck and Embedding Representation of Speech for DNN-Based Language and Speaker Recognition*. Thèse de doctorat, Universidad Autónoma de Madrid.

[Lozano-Diez *et al.*, 2018a] Lozano-Diez, A., Plchot, O., Matějka, P. et Gonzalez-Rodriguez, J. (2018a). DNN based embeddings for language recognition. *In Proceedings of ICASSP 2018 - IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5184–5188, Calgary, AB. IEEE.

[Lozano-Diez *et al.*, 2018b] Lozano-Diez, A., Plchot, O., Matějka, P., Novotný, O. et Gonzalez-Rodriguez, J. (2018b). Analysis of DNN-based embeddings for language recognition on the NIST LRE 2017. *In Proceedings of Odyssey 2018 - The Speaker and Language Recognition Workshop*, pages 39–46. ISCA.

[Lozano-Diez *et al.*, 2017] Lozano-Diez, A., Zazo, R., Toledano, D. T. et Gonzalez-Rodriguez, J. (2017). An analysis of the influence of deep neural network (DNN) topology in bottleneck feature based language recognition. *In PloS one*, volume 12, page e0182580. Public Library of Science San Francisco, CA USA.

[Lozano-Diez *et al.*, 2015] Lozano-Diez, A., Zazo-Candil, R., Gonzalez-Dominguez, J., Toledano, D. T. et Gonzalez-Rodriguez, J. (2015). An end-to-end approach to language identification in short utterances using convolutional neural networks. *In Proceedings of Interspeech 2015 - Annual Conference of the International Speech Communication Association*.

[Lu *et al.*, 2021] Lu, X., Shen, P., Tsao, Y. et Kawai, H. (2021). Unsupervised neural adaptation model based on optimal transport for spoken language identification. *In Proceedings of ICASSP 2021 - IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 7213–7217, Toronto, ON, Canada. IEEE.

[Lyu *et al.*, 2013] Lyu, D.-C., Chng, E.-S. et Li, H. (2013). Language diarization for code-switch conversational speech. *In Proceedings of ICASSP 2013 - IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 7314–7318.

[Mansour *et al.*, 2008] Mansour, Y., Mohri, M. et Rostamizadeh, A. (2008). Domain adaptation with multiple sources. *In Advances in neural information processing systems*, volume 21, pages 1041–1048.

[Martinez *et al.*, 2011] Martinez, D., Plchot, O., Burget, L., Glembek, O. et Matějka, P. (2011). Language recognition in i-vectors space. *In Proceedings of Interspeech 2011 - Annual Conference of the International Speech Communication Association*.

[Matějka *et al.*, 2014] Matějka, P., Zhang, L., Ng, T., Mallidi, S. H., Glembek, O., Ma, J. et Zhang, B. (2014). Neural network bottleneck features for language identification. *In Proceedings of Odyssey 2014 - The Speaker and Language Recognition Workshop*, pages 299–304.

[Mateju *et al.*, 2018] Mateju, L., Cerva, P., Zdansky, J. et Safarik, R. (2018). Using deep neural networks for identification of slavic languages from acoustic signal. *In Proceedings of Interspeech 2018 - Annual Conference of the International Speech Communication Association*, pages 1803–1807. ISCA.

[Matrouf *et al.*, 2011] Matrouf, D., Verdet, F., Rouvier, M., Bonastre, J.-F. et Linarès, G. (2011). Modeling nuisance variabilities with factor analysis for GMM-based audio pattern classification. *In Computer Speech & Language*, volume 25, pages 481–498.

[Mccree *et al.*, 2016] Mccree, A., Sell, G. et Garcia-Romero, D. (2016). Augmented data training of joint acoustic/phonotactic DNN i-vectors for NIST LRE15. *In Proceedings of Odyssey 2016 - The Speaker and Language Recognition Workshop*, pages 204–209.

[Mccree *et al.*, 2018] Mccree, A., Snyder, D., Sell, G. et Garcia-Romero, D. (2018). Language recognition for telephone and video speech : the JHU HLTCOE submission for NIST LRE17. *In Proceedings of Odyssey 2018 - The Speaker and Language Recognition Workshop*, pages 68–73. ISCA.

[McLaren *et al.*, 2016a] McLaren, M., Castán, D. et Ferrer, L. (2016a). Analyzing the effect of channel mismatch on the SRI Language Recognition Evaluation 2015 system. *In Proceedings of Odyssey 2016 - The Speaker and Language Recognition Workshop*, pages 188–195.

[McLaren *et al.*, 2018a] McLaren, M., Castán, D., Nandwana, M. K., Ferrer, L. et Yilmaz, E. (2018a). How to train your speaker embeddings extractor. *In Proceedings of Odyssey 2018 - The Speaker and Language Recognition Workshop*, pages 327–334. ISCA.

[McLaren *et al.*, 2017] McLaren, M., Ferrer, L., Castan, D. et Lawson, A. (2017). Calibration approaches for language detection. *In Proceedings of Interspeech 2017 - Annual Conference of the International Speech Communication Association*, pages 2804–2808. ISCA.

[McLaren *et al.*, 2016b] McLaren, M., Ferrer, L. et Lawson, A. (2016b). Exploring the role of phonetic bottleneck features for speaker and language recognition. *In Proceedings of ICASSP 2016 - IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5575–5579. IEEE.

[McLaren *et al.*, 2018b] McLaren, M., Nandwana, M. K., Castán, D. et Ferrer, L. (2018b). Approaches to multi-domain language recognition. *In Proceedings of Odyssey 2018 - The Speaker and Language Recognition Workshop*, pages 90–97.

[McLaren *et al.*, 2020] McLaren, M., Rahman, M. H., Castan, D., Nandwana, M. K. et Lawson, A. (2020). Adaptive mean normalization for unsupervised adaptation of speaker embeddings. *In Proceedings of Odyssey 2020 - The Speaker and Language Recognition Workshop*, pages 88–94. ISCA.

[Meng *et al.*, 2019] Meng, Z., Zhao, Y., Li, J. et Gong, Y. (2019). Adversarial speaker verification. *In Proceedings of ICASSP 2019 - IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6216–6220.

[Miao *et al.*, 2019] Miao, X., McLoughlin, I. et Yan, Y. (2019). A new time-frequency attention mechanism for TDNN and CNN-LSTM-TDNN, with application to language identification. *In Proceedings of Interspeech 2019 - Annual Conference of the International Speech Communication Association*, pages 4080–4084. ISCA.

[Mingote *et al.*, 2019] Mingote, V., Castan, D., McLaren, M., Nandwana, M. K., Ortega, E. L. et Miguel, A. (2019). Language recognition using triplet neural networks. *In Proceedings of Interspeech 2019 - Annual Conference of the International Speech Communication Association*, pages 4025–4029.

[Misra et Hansen, 2018] Misra, A. et Hansen, J. H. L. (2018). Modelling and compensation for language mismatch in speaker verification. *In Speech Communication*, volume 96, pages 58–66.

[Misra *et al.*, 2016] Misra, A., Zhang, Q., Kelly, F. et Hansen, J. H. (2016). Between-class covariance correction for linear discriminant analysis in language recognition. *In Proceedings of Odyssey 2016 - The Speaker and Language Recognition Workshop*, pages 68–73.

[Mohammadamini *et al.*, 2020] Mohammadamini, M., Matrouf, D. et Noé, P.-G. (2020). Denoising x-vectors for robust speaker recognition. *In Proceedings of Odyssey 2020 - The Speaker and Language Recognition Workshop*, pages 75–80. ISCA.

[Molau *et al.*, 2003] Molau, S., Hilger, F. et Ney, H. (2003). Feature space normalization in adverse acoustic conditions. *In Proceedings of ICASSP 2003 - IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 1, pages I–I.

[Monteiro *et al.*, 2019] Monteiro, J., Alam, J. et Falk, T. H. (2019). Residual convolutional neural network with attentive feature pooling for end-to-end language identification from short-duration speech. *In Computer Speech & Language*, volume 58, pages 364–376.

[Muandet *et al.*, 2013] Muandet, K., Balduzzi, D. et Schölkopf, B. (2013). Domain generalization via invariant feature representation. *In Proceedings of ICML 2013 - International Conference on Machine Learning*, pages 10–18. PMLR.

[Nautsch *et al.*, 2016] Nautsch, A., Saeidi, R., Rathgeb, C. et Busch, C. (2016). Robustness of quality-based score calibration of speaker recognition systems with respect to low-SNR and short-duration conditions. *In Proceedings of Odyssey 2016 - The Speaker and Language Recognition Workshop*, pages 358–365.

[Nercessian *et al.*, 2016] Nercessian, S., Torres-Carrasquillo, P. et Martinez-Montes, G. (2016). Approaches for language identification in mismatched environments. *In Proceedings of SLT 2016 - IEEE Spoken Language Technology Workshop*, pages 335–340. IEEE.

[NIST, 2007] NIST (2007). The 2007 NIST language recognition evaluation plan. *In http://www.nist.gov/speech/tests/lre/2007/LRE07EvalPlan-v8b.pdf*.

[NIST, 2013] NIST (2013). LRE 2011 results. *In https://www.nist.gov/itl/iad/mig/lre11-results*.

[Novotný *et al.*, 2018] Novotný, O., Plchot, O., Matějka, P., Mošner, L. et Glembek, O. (2018). On the use of x-vectors for robust speaker recognition. *In Proceedings of Odyssey 2018 - The Speaker and Language Recognition Workshop*, pages 168–175. ISCA.

[Padi *et al.*, 2019a] Padi, B., Mohan, A. et Ganapathy, S. (2019a). Attention based hybrid i-vector BLSTM model for language recognition. *In Proceedings of Interspeech 2019 - Annual Conference of the International Speech Communication Association*, pages 1263–1267. ISCA.

[Padi *et al.*, 2019b] Padi, B., Mohan, A. et Ganapathy, S. (2019b). End-to-end language recognition using attention based hierarchical gated recurrent unit models. *In Proceedings of ICASSP 2019 - IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5966–5970, Brighton, United Kingdom. IEEE.

[Padi *et al.*, 2018] Padi, B., Ramoji, S., Yeruva, V., Kumar, S. et Ganapathy, S. (2018). The LEAP language recognition system for LRE 2017 challenge-improvements and error analysis. *In Proceedings of Odyssey 2018 - The Speaker and Language Recognition Workshop*, pages 31–38.

[Panayotov *et al.*, 2015] Panayotov, V., Chen, G., Povey, D. et Khudanpur, S. (2015). Librispeech : an ASR corpus based on public domain audio books. *In Proceedings of ICASSP 2015 - IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5206–5210, South Brisbane, Queensland, Australia. IEEE.

[Park *et al.*, 2019] Park, D. S., Chan, W., Zhang, Y., Chiu, C.-C., Zoph, B., Cubuk, E. D. et Le, Q. V. (2019). Specaugment : a simple data augmentation method for automatic speech recognition. *In Proceedings of Interspeech 2019 - Annual Conference of the International Speech Communication Association*, pages 2613–2617.

[Pascual *et al.*, 2019] Pascual, S., Ravanelli, M., Serrà, J., Bonafonte, A. et Bengio, Y. (2019). Learning problem-agnostic speech representations from multiple self-

supervised tasks. *In Proceedings of Interspeech 2019 - Annual Conference of the International Speech Communication Association*, pages 161–165.

[Pedregosa *et al.*, 2011] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. et Duchesnay, E. (2011). Scikit-learn : machine learning in Python. *In Journal of Machine Learning Research*, volume 12, pages 2825–2830.

[Peng *et al.*, 2019] Peng, Z., Feng, S. et Lee, T. (2019). Adversarial multi-task deep features and unsupervised back-end adaptation for language recognition. *In Proceedings of ICASSP 2019 - IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5961–5965. IEEE.

[Peyré *et al.*, 2019] Peyré, G., Cuturi, M. *et al.* (2019). Computational optimal transport. *In Foundations and Trends® in Machine Learning*, volume 11, pages 355–607. Now Publishers, Inc.

[Plchot *et al.*, 2016] Plchot, O., Matějka, P., Glembek, O., Fér, R., Novotný, O., Pešán, J., Burget, L., Brümmer, N. et Cumani, S. (2016). BAT system description for NIST LRE 2015. *In Proceedings of Odyssey 2016 - The Speaker and Language Recognition Workshop*, pages 166–173.

[Plchot *et al.*, 2018] Plchot, O., Matějka, P., Novotný, O., Cumani, S., Lozano-Diez, A., Slavicek, J., Diez, M., Grézl, F., Glembek, O., Kamsali, M. *et al.* (2018). Analysis of BUT-PT submission for NIST LRE 2017. *In Proceedings of Odyssey 2018 - The Speaker and Language Recognition Workshop*, pages 47–53.

[Plchot *et al.*, 2017] Plchot, O., Matějka, P., Silnova, A., Novotný, O., Sánchez, M. D., Rohdin, J., Glembek, O., Brümmer, N., Swart, A., Jorrín-Prieto, J., García, P., Buera, L., Kenny, P., Alam, J. et Bhattacharya, G. (2017). Analysis and description of ABC submission to NIST SRE 2016. *In Proceedings of Interspeech 2017 - Annual Conference of the International Speech Communication Association*, pages 1348–1352. ISCA.

[Pratap *et al.*, 2020] Pratap, V., Sriram, A., Tomasello, P., Hannun, A., Liptchinsky, V., Synnaeve, G. et Collobert, R. (2020). Massively multilingual ASR : 50 languages, 1 model, 1 billion parameters. *In Proceedings of Interspeech 2020 - Annual Conference of the International Speech Communication Association*, pages 4751–4755.

[Punjabi *et al.*, 2021] Punjabi, S., Arsikere, H., Raeesy, Z., Chandak, C., Bhave, N., Bansal, A., Müller, M., Murillo, S., Rastrow, A., Stolcke, A. *et al.* (2021). Joint ASR and language identification using RNN-T : an efficient approach to dynamic language switching. *In Proceedings of ICASSP 2021 - IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 7218–7222. IEEE.

[Rahman *et al.*, 2018] Rahman, M. H., Himawan, I., Dean, D., Fookes, C. et Sridharan, S. (2018). Domain-invariant i-vector feature extraction for PLDA speaker verification. *In Proceedings of Odyssey 2018 - The Speaker and Language Recognition Workshop*, pages 155–161. ISCA.

[Raj *et al.*, 2019] Raj, D., Snyder, D., Povey, D. et Khudanpur, S. (2019). Probing the information encoded in x-vectors. *In Proceedings of ASRU 2019 - IEEE Automatic Speech Recognition and Understanding Workshop*, pages 726–733. IEEE.

[Ramos *et al.*, 2018] Ramos, D., Franco-Pedroso, J., Lozano-Diez, A. et Gonzalez-Rodriguez, J. (2018). Deconstructing cross-entropy for probabilistic binary classifiers. *In Entropy*, volume 20.

[Ravanelli et Bengio, 2018] Ravanelli, M. et Bengio, Y. (2018). Speaker Recognition from raw waveform with SincNet. *In Proceedings of SLT 2018 - IEEE Spoken Language Technology Workshop*, pages 1021–1028.

[Redko *et al.*, 2020] Redko, I., Morvant, E., Habrard, A., Sebban, M. et Bennani, Y. (2020). A survey on domain adaptation theory : Learning bounds and theoretical guarantees. *In arXiv e-prints*, volume 2004, page arXiv :2004.11829.

[Ren *et al.*, 2019] Ren, Z., Yang, G. et Xu, S. (2019). Two-stage training for Chinese dialect recognition. *In Proceedings of Interspeech 2019 - Annual Conference of the International Speech Communication Association*, pages 4050–4054.

[Richardson *et al.*, 2016] Richardson, F., Nemsick, B. et Reynolds, D. (2016). Channel compensation for speaker recognition using MAP adapted PLDA and denoising DNNs. *In Proceedings of Odyssey 2016 - The Speaker and Language Recognition Workshop*, pages 225–230.

[Richardson *et al.*, 2018] Richardson, F., Torres-Carrasquillo, P. A., Borgstrom, J., Sturim, D. E., Gwon, Y., Villalba, J., Trmal, J., Chen, N., Dehak, R. et Dehak, N. (2018). The MIT Lincoln Laboratory/JHU/EPITA-LSE LRE17 system. *In Proceedings of Odyssey 2020 - The Speaker and Language Recognition Workshop*, pages 54–59.

[Rohdin *et al.*, 2019] Rohdin, J., Stafylakis, T., Silnova, A., Zeinali, H., Burget, L. et Plchot, O. (2019). Speaker verification using end-to-end adversarial language adaptation. *In Proceedings of ICASSP 2019 - IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6006–6010. IEEE.

[Rouvier *et al.*, 2021a] Rouvier, M., Bousquet, P.-M. et Duret, J. (2021a). Study on the temporal pooling used in deep neural networks for speaker verification. *In arXiv :2105.04310 [cs, eess]*.

[Rouvier *et al.*, 2021b] Rouvier, M., Dufour, R. et Bousquet, P.-M. (2021b). Review of different robust x-vector extractors for speaker verification. *In Proceedings of EUSIPCO 2020 - European Signal Processing Conference*, pages 1–5, Amsterdam, Netherlands. IEEE.

[Sadjadi *et al.*, 2018a] Sadjadi, S. O., Kheyrkhah, T., Greenberg, C., Singer, E., Reynolds, D., Mason, L. et Hernandez-Cordero, J. (2018a). Performance analysis of the 2017 NIST Language Recognition Evaluation. *In Proceedings of Interspeech 2018 - Annual Conference of the International Speech Communication Association*, pages 1798–1802. ISCA.

[Sadjadi *et al.*, 2018b] Sadjadi, S. O., Kheyrkhah, T., Tong, A., Greenberg, C. S., Reynolds, D. A., Singer, E., Mason, L. P. et Hernandez-Cordero, J. (2018b). The 2017 NIST Language Recognition Evaluation. *In Proceedings of Odyssey 2018 - The Speaker and Language Recognition Workshop*, pages 82–89.

[Sarthak et Mittal, 2019] Sarthak, S. S. B. et Mittal, G. (2019). Spoken language identification using convnets. *In Ambient Intelligence*, volume 11912, page 252. Springer Nature.

[Seltzer *et al.*, 2013] Seltzer, M. L., Yu, D. et Wang, Y. (2013). An investigation of deep neural networks for noise robust speech recognition. *In Proceedings of ICASSP 2013 - IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 7398–7402, Vancouver, BC, Canada. IEEE.

[Shen *et al.*, 2018] Shen, P., Lu, X., Li, S. et Kawai, H. (2018). Feature representation of short utterances based on knowledge distillation for spoken language identification. *In Proceedings of Interspeech 2018 - Annual Conference of the International Speech Communication Association*, pages 1813–1817. ISCA.

[Shen *et al.*, 2020] Shen, P., Lu, X., Sugiura, K., Li, S. et Kawai, H. (2020). Compensation on x-vector for short utterance spoken language identification. *In Proceedings of Odyssey 2020 - The Speaker and Language Recognition Workshop*, pages 47–52.

[Shi *et al.*, 2020] Shi, Y., Chen, M., Huang, Q. et Hain, T. (2020). T-vectors : weakly supervised speaker identification using hierarchical transformer model. *In arXiv :2010.16071 [cs, eess].*

[Shlens, 2014] Shlens, J. (2014). A tutorial on principal component analysis. *In arXiv :1404.1100 [cs, stat].*

[Shon *et al.*, 2019] Shon, S., Tang, H. et Glass, J. (2019). VoiceID loss : speech enhancement for speaker verification. *In Proceedings of Interspeech 2019 - Annual Conference of the International Speech Communication Association*, pages 2888–2892.

[Silnova *et al.*, 2018] Silnova, A., Matějka, P., Glembek, O., Plchot, O., Novotný, O., Grézl, F., Schwarz, P., Burget, L. et Černocký, J. H. (2018). BUT/-Phonexia bottleneck feature extractor. *In Proceedings of Odyssey 2018 - The Speaker and Language Recognition Workshop*, pages 283–287. ISCA.

[Singer *et al.*, 2003] Singer, E., Torres-Carrasquillo, P. A., Gleason, T. P., Campbell, W. M. et Reynolds, D. A. (2003). Acoustic, phonetic, and discriminative approaches to automatic language identification. *In Proceedings of the Eighth European conference on speech communication and technology.*

[Snyder, 2020] Snyder, D. (2020). *X-vectors : robust neural embeddings for speaker recognition.* Thèse de doctorat, Johns Hopkins University.

[Snyder *et al.*, 2015] Snyder, D., Chen, G. et Povey, D. (2015). MUSAN : a music, speech, and noise corpus. *In arXiv :1510.08484 [cs].*

[Snyder *et al.*, 2018] Snyder, D., Garcia-Romero, D., McCree, A., Sell, G., Povey, D. et Khudanpur, S. (2018). Spoken language recognition using x-vectors. *In*

*Proceedings of Odyssey 2018 - The Speaker and Language Recognition Workshop*, pages 105–111.

[Snyder *et al.*, 2016] Snyder, D., Ghahremani, P., Povey, D., Garcia-Romero, D., Carmiel, Y. et Khudanpur, S. (2016). Deep neural network-based speaker embeddings for end-to-end speaker verification. *In Proceedings of SLT 2016 - IEEE Spoken Language Technology Workshop*, pages 165–170. IEEE.

[Sohn, 2016] Sohn, K. (2016). Improved deep metric learning with multi-class n-pair loss objective. *In Advances in neural information processing systems*, pages 1857–1865.

[Solomonoff *et al.*, 2007] Solomonoff, A., Campbell, W. M. et Quillen, C. (2007). Nuisance attribute projection. *In Speech Communication*, pages 1–73. Elsevier Science BV.

[Srivastava *et al.*, 2014] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. et Salakhutdinov, R. (2014). Dropout : a simple way to prevent neural networks from overfitting. *In The Journal of Machine Learning Research*, volume 15, pages 1929–1958. JMLR. org.

[Stevens *et al.*, 1937] Stevens, S. S., Volkmann, J. et Newman, E. B. (1937). A scale for the measurement of the psychological magnitude pitch. *In The journal of the acoustical society of America*, volume 8, pages 185–190. Acoustical Society of America.

[Sturim *et al.*, 2016] Sturim, D. E., Richardson, F. S., Singer, E., Godoy, E. C., Reynolds, D. A., Torres-Carrasquillo, P. A., Dehak, N. et Shum, S. (2016). The MITLL NIST LRE 2015 language recognition system. Rapport technique, MIT Lincoln Laboratory Lexington United States.

[Sun et Saenko, 2016] Sun, B. et Saenko, K. (2016). Deep CORAL : Correlation alignment for deep domain adaptation. *In Proceedings of ECCV 2016 - European Conference on Computer Vision*, pages 443–450. Springer.

[Sun *et al.*, 2015] Sun, S., Shi, H. et Wu, Y. (2015). A survey of multi-source domain adaptation. *In Information Fusion*, volume 24, pages 84–92. Elsevier.

[Synnaeve *et al.*, 2020] Synnaeve, G., Xu, Q., Kahn, J., Likhomanenko, T., Grave, E., Pratap, V., Sriram, A., Liptchinsky, V. et Collobert, R. (2020). End-to-end ASR : from supervised to semi-supervised learning with modern architectures. *In arXiv :1911.08460 [cs, eess]*.

[Talkin et Kleijn, 1995] Talkin, D. et Kleijn, W. B. (1995). A robust algorithm for pitch tracking (RAPT). *In Speech coding and synthesis*, volume 495, page 518.

[Tang *et al.*, 2017a] Tang, Z., Wang, D., Chen, Y. et Chen, Q. (2017a). AP17-OLR challenge : data, plan, and baseline. *In Proceedings of APSIPA ASC 2017 - Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, pages 749–753. IEEE.

[Tang *et al.*, 2017b] Tang, Z., Wang, D., Chen, Y., Li, L. et Abel, A. (2017b). Phonetic temporal neural model for language identification. *In IEEE/ACM Transactions on Audio, Speech, and Language Processing*, volume 26, pages 134–144. IEEE.

[Tang *et al.*, 2019] Tang, Z., Wang, D. et Song, L. (2019). AP19-OLR challenge : three tasks and their baselines. *In Proceedings of APSIPA ASC 2019 - Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, pages 1917–1921. IEEE.

[Tharwat *et al.*, 2017] Tharwat, A., Gaber, T., Ibrahim, A. et Hassanien, A. E. (2017). Linear discriminant analysis : a detailed tutorial. *In AI Communications*, volume 30, pages 169–190.

[Thomas *et al.*, 2014] Thomas, S., Ganapathy, S., Saon, G. et Soltau, H. (2014). Analyzing convolutional neural networks for speech activity detection in mismatched acoustic conditions. *In Proceedings of ICASSP 2014 - IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 2519–2523, Florence, Italy. IEEE.

[Tian *et al.*, 2016] Tian, Y., He, L., Liu, Y. et Liu, J. (2016). Investigation of senone-based long-short term memory RNNs for spoken language recognition. *In Proceedings of Odyssey 2016 - The Speaker and Language Recognition Workshop*, pages 89–93.

[Titus *et al.*, 2020] Titus, A., Silovsky, J., Chen, N., Hsiao, R., Young, M. et Ghoshal, A. (2020). Improving language identification for multilingual speakers. *In Proceedings of ICASSP 2020 - IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8284–8288. IEEE.

[Tong *et al.*, 2016] Tong, A., Greenberg, C., Martin, A., Banse, D., Howard, J., Zhao, H., Doddington, G., Garcia-Romero, D., McCree, A., Reynolds, D., Singer, E., Hernandez-Cordero, J. et Mason, L. (2016). Summary of the 2015 NIST language recognition i-vector machine learning challenge. *In Proceedings of Odyssey 2016 - The Speaker and Language Recognition Workshop*, pages 297–302.

[Torres-Carrasquillo *et al.*, 2002] Torres-Carrasquillo, P. A., Singer, E., Kohler, M. A., Greene, R. J., Reynolds, D. A. et Deller Jr, J. R. (2002). Approaches to language identification using Gaussian mixture models and shifted delta cepstral features. *In Proceedings of Interspeech 2002 - Annual Conference of the International Speech Communication Association*, pages 89–92.

[Trong *et al.*, 2018] Trong, T. N., Hautamaki, V. et Jokinen, K. (2018). Staircase network : structural language identification via hierarchical attentive units. *In Proceedings of Odyssey 2018 - The Speaker and Language Recognition Workshop*, pages 60–67. ISCA.

[Trong *et al.*, 2016] Trong, T. N., Hautamäki, V. et Lee, K.-A. (2016). Deep language : a comprehensive deep learning approach to end-to-end language recognition. *In Proceedings of Odyssey 2016 - The Speaker and Language Recognition Workshop*, pages 109–116.

[Van Der Maaten et Hinton, 2008] Van Der Maaten, L. et Hinton, G. (2008). Visualizing data using t-SNE. *In Journal of Machine Learning Research*.

[Vapnik, 1992] Vapnik, V. (1992). Principles of risk minimization for learning theory. *In Advances in Neural Information Processing Systems*, pages 831–838.

[Vapnik, 2013] Vapnik, V. (2013). *The Nature of Statistical Learning Theory*. Springer Science & Business Media.

[Vaswani *et al.*, 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł. et Polosukhin, I. (2017). Attention is all you need. *In Advances in neural information processing systems*, pages 5998–6008.

[Verdet, 2011] Verdet, F. (2011). *Exploring Variabilities through Factor Analysis in Automatic Acoustic Language Recognition*. Thèse de doctorat, Université d'Avignon.

[Verdet *et al.*, 2010] Verdet, F., Matrouf, D., Bonastre, J.-F. et Hennebert, J. (2010). Coping with two different transmission channels in language recognition. *In Proceedings of Odyssey 2010 - The Speaker and Language Recognition Workshop*, page 39.

[Villalba *et al.*, 2020] Villalba, J., Chen, N., Snyder, D., Garcia-Romero, D., McCree, A., Sell, G., Borgstrom, J., García-Perera, L. P., Richardson, F., Dehak, R., Torres-Carrasquillo, P. A. et Dehak, N. (2020). State-of-the-art speaker recognition with neural network embeddings in NIST SRE18 and Speakers in the Wild evaluations. *In Computer Speech & Language*, volume 60, page 101026.

[Wan *et al.*, 2019] Wan, L., Sridhar, P., Yu, Y., Wang, Q. et Moreno, I. L. (2019). Tuplemax loss for language identification. *In Proceedings of ICASSP 2019 - IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5976–5980. IEEE.

[Wang *et al.*, 2016] Wang, D., Li, L., Tang, D. et Chen, Q. (2016). AP16-OL7 : a multilingual database for oriental languages and a language recognition baseline. *In Proceedings of APSIPA ASC 2016 - Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, pages 1–5. IEEE.

[Wang *et al.*, 2021a] Wang, J., Lan, C., Liu, C., Ouyang, Y., Zeng, W. et Qin, T. (2021a). Generalizing to unseen domains : a survey on domain generalization. *In arXiv :2103.03097 [cs]*.

[Wang *et al.*, 2021b] Wang, S., Yang, Y., Qian, Y. et Yu, K. (2021b). Revisiting the statistics pooling layer in deep speaker embedding learning. *In Proceedings of ISCSLP 2021 - International Symposium on Chinese Spoken Language Processing*, pages 1–5, Hong Kong. IEEE.

[Wang *et al.*, 2019] Wang, Y., Chen, T., Xu, H., Ding, S., Lv, H., Shao, Y., Peng, N., Xie, L., Watanabe, S. et Khudanpur, S. (2019). Espresso : a fast end-to-end neural speech recognition toolkit. *In Proceedings of ASRU 2019 - IEEE Automatic Speech Recognition and Understanding Workshop*, pages 136–143. IEEE.

[Wilson et Cook, 2020] Wilson, G. et Cook, D. J. (2020). A survey of unsupervised deep domain adaptation. *In ACM Transactions on Intelligent Systems and Technology (TIST)*, volume 11, pages 1–46. ACM New York, NY, USA.

[Xia *et al.*, 2019] Xia, W., Huang, J. et Hansen, J. H. (2019). Cross-lingual text-independent speaker verification using unsupervised adversarial discriminative domain

adaptation. *In Proceedings of ICASSP 2019 - IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5816–5820.

[Yu et Deng, 2015] Yu, D. et Deng, L. (2015). *Automatic Speech Recognition*. Signals and Communication Technology. Springer London, London.

[Zazo *et al.*, 2016a] Zazo, R., Lozano-Diez, A., Gonzalez-Dominguez, J., Toledano, D. T. et Gonzalez-Rodriguez, J. (2016a). Language identification in short utterances using long short-term memory (LSTM) recurrent neural networks. *In PLOS ONE*, volume 11, page e0146917. Public Library of Science.

[Zazo *et al.*, 2016b] Zazo, R., Lozano-Diez, A. et Gonzalez-Rodriguez, J. (2016b). Evaluation of an LSTM-RNN system in different NIST language recognition frameworks. *In Proceedings of Odyssey 2016 - The Speaker and Language Recognition Workshop*, pages 231–236.

[Zeghidour *et al.*, 2020] Zeghidour, N., Teboul, O., de Chaumont Quitry, F. et Tagliasacchi, M. (2020). LEAF : a learnable frontend for audio classification. *In Proceedings of ICLR 2020 - International Conference on Learning Representations*.

[Zhang et Koishida, 2017] Zhang, C. et Koishida, K. (2017). End-to-end text-independent speaker verification with triplet loss on short utterances. *In Proceedings of Interspeech 2017 - Annual Conference of the International Speech Communication Association*, pages 1487–1491.

[Zhang *et al.*, 2018] Zhang, C., Ranjan, S. et Hansen, J. (2018). An analysis of transfer learning for domain mismatched text-independent speaker verification. *In Proceedings of Odyssey 2018 - The Speaker and Language Recognition Workshop*, pages 181–186. ISCA.

[Zhang et Hansen, 2017] Zhang, Q. et Hansen, J. H. (2017). Dialect recognition based on unsupervised bottleneck features. *In Proceedings of Interspeech 2017 - Annual Conference of the International Speech Communication Association*, pages 2576–2580. ISCA.

[Zhao *et al.*, 2016] Zhao, H., Bansé, D., Doddington, G., Greenberg, C., Hernández-Cordero, J., Howard, J., Mason, L., Martin, A., Reynolds, D., Singer, E. et Tong, A. (2016). Results of the 2015 NIST Language Recognition Evaluation. *In Proceedings of Interspeech 2016 - Annual Conference of the International Speech Communication Association*, pages 3206–3210.

[Zhao *et al.*, 2019] Zhao, M., Li, R., Yan, S., Li, Z., Lu, H., Xia, S., Hong, Q. et Li, L. (2019). Phone-aware multi-task learning and length expanding for short-duration language recognition. *In Proceedings of 2019 APSIPA ASC - Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, pages 433–437. IEEE.

[Zhiyuan Tang, 2018] Zhiyuan Tang, Dong Wang, Q. C. (2018). AP18-OLR challenge : three tasks and their baselines. *In Proceedings of APSIPA ASC 2018 - Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*. IEEE.

[Zhou *et al.*, 2021] Zhou, K., Liu, Z., Qiao, Y., Xiang, T. et Loy, C. C. (2021). Domain generalization : a survey. *In arXiv :2103.02503 [cs]*.

[Zissman, 1996] Zissman, M. (1996). Comparison of four approaches to automatic language identification of telephone speech. *In IEEE Transactions on Speech and Audio Processing*, volume 4, page 31.

# Résumé

La tâche de reconnaissance de la langue consiste à prédire la langue utilisée dans un énoncé audio contenant de la parole. Depuis 2017, les systèmes atteignant les meilleures performances reposent sur un réseau de neurones profond, entraîné à identifier la langue pour l'ensemble du segment. Ces systèmes subissent une perte de performance lorsqu'ils sont exposés à une variation des canaux de transmission entre les données d'entraînement et d'évaluation. L'objet de cette thèse est l'exploration d'approches permettant de limiter cette perte de performance dans le cadre de ces nouveaux systèmes. Nos travaux peuvent être regroupés en trois directions : l'étude d'une méthode d'amélioration de la robustesse au canal des systèmes, l'analyse de leur robustesse et la simplification de la recette d'apprentissage.

Une augmentation de l'invariance, par rapport au canal de transmission, des représentations utilisées par le réseau de neurones peut augmenter la robustesse du système. Nous montrons que la régularisation de la fonction de coût utilisée lors de l'entraînement du réseau de neurones est un outil efficace pour augmenter cette invariance. Deux types de fonction de régularisation sont analysés. Les mesures de divergence entre les domaines réduisent efficacement la variabilité entre des canaux identifiés, elles peuvent également être utilisées pour valoriser des données non annotées dans le cadre d'un apprentissage semi-supervisé. Les fonctions de coût de *metric learning* permettent de réduire des variabilités inconnues dans l'ensemble d'apprentissage. Nous montrons comment cette méthode peut être mise en œuvre dans trois scénarios d'apprentissage d'intérêt pratique : l'adaptation de domaine non supervisée, l'apprentissage multi-domaines et la généralisation à un domaine inconnu.

Au cours de l'étude de cette approche, nous développons des méthodes d'analyse de la qualité des représentations. Elles visent à mesurer la variabilité des représentations due au canal de transmission et à la comparer à la variabilité due à la langue. Deux outils sont introduits : le calcul de rapports entre les covariance inter-classes et intra-classes et la mesure de divergences entre groupes de représentations. Ceux-ci nous permettent d'évaluer quantitativement la robustesse des représentations au changement de canal et donc de comprendre l'effet des fonctions de régularisation sur l'espace des représentations. En particulier, ces méthodes révèlent que l'augmentation de l'invariance entre les canaux peut mener à des représentations plus discriminantes entre les langues et donc à une amélioration de la performance sur chacun des canaux de transmission.

Enfin, nous contribuons à l'amélioration de la recette d'entraînement d'un autre module du système, l'extracteur de *bottleneck features*. Nous montrons qu'un réseau de neurones de reconnaissance de la parole de bout en bout multilingue permet de réaliser cette extraction, avec une meilleure performance et une recette d'apprentissage simplifiée. L'utilisation d'augmentations de données et de méthodes de régularisation améliore la performance de ce module. D'autre part nous montrons qu'un gain de performance peut

être obtenu en réalisant un entraînement conjoint de ce module avec le réseau d'identification de la langue. Cela ouvre la voie à l'application simultanée des fonctions de régularisation étudiées aux deux modules.

**Mots-clés:** reconnaissance de la langue, adaptation de domaine, réseau de neurones, canal de transmission, robustesse, représentations

# Abstract

Language recognition is the task of predicting the language used in a test speech utterance. Since 2017, the best performing systems have been based on a deep neural network which is trained to predict language labels for the whole utterance. These systems suffer from a drop in performance when they are exposed to a change of the transmission channel between train and test data. The goal of this thesis is to investigate approaches to limit this performance drop, for these new systems.

An increase in the invariance, with respect to the transmission channel, of the representations used by the neural network can increase the robustness of the system. We show that the regularization of the loss function used to train the neural network is an efficient approach to increase invariance. Two kinds of regularization functions are analysed. Divergence measures between domains reduce effectively the variability between known domains, they can also be used to incorporate unlabeled data into the training set in a semi-supervised learning framework. Metric learning cost functions are able to reduce unknown variabilities within the training set. We show how this regularization method can be enforced for three practical learning settings : unsupervised domain adaptation, multi-domain learning and domain generalization.

During this work, we have designed methods for analyzing the quality of the representations. They aim at evaluating the variability of the representations induced by the transmission channel and to compare it to the variability that caused the language. Two tools are proposed : ratio between inter class and intra class covariance matrices and divergence measures between groups of representations. With these tools, we quantitatively evaluate the robustness to a change of transmission channel of the representations and analyse the effect of the regularization functions over the space of representations. We understand that an increase in invariance between channels can lead to more discriminative representations between languages and consequently to an increase in performance over each transmission channel.

Finally, we contribute to the improvement of the training recipe of another module of the system, the bottleneck feature extractor. We replace it with a multilingual end-to-end automatic speech recognition neural network. It achieves a simiar performance as a traditional bottleneck feature extractor with a simplified training recipe. The use of data augmentation and regularization methods improves further this module. Moreover we show that a performance gain can be achieved with a joint training of the bottleneck feature extractor along with the language identification neural network. This paves the way to the application of the proposed regularization loss functions to the two modules jointly.



**Keywords:** language recognition, domain adaptation, neural network, transmission channel, robustness, representations