Robustness of DNN-based speaker recognition systems against environmental variabilities

Mohammad Mohammadamini

AVIGNON UNIVERSITÉ

THÈSE DE DOCTORAT D'AVIGNON UNIVERSITÉ

École Doctorale n°536
Agrosciences et Sciences

Mention de doctorat :
Informatique

LIA (Laboratoire informatique d'Avignon)

Présentée par
Mohammad MOHAMMADAMINI

Robustness of DNN-based speaker recognition systems against environmental variabilities

---

Soutenue publiquement le 15/05/2023 devant le jury composé de :

PR. LARCHER Anthony	Rapporteur	Le Mans Université
DR. MATEJKA Pavel	Rapporteur	Senior researcher at Brno University
DR. ILLINA Irina	Examiner	Associate Professor at Université de Lorraine-INRIA
DR. BARRAS Claude	Examiner	Vocapia Research, on leave Université Paris-Saclay
PR. BONASTRE Jean-Francois	Examiner	Avignon Université
DR. ROUVIER Mickael	Examiner	Associate Professor at Avignon Université
DR. MATROUF Driss	Supervisor	Associate Professor at Avignon Université
PR. LEFEVRE Fabrice	President	Avignon Université

LABORATOIRE INFORMATIQUE D'AVIGNON

TO Xanî, Nalî

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude and appreciation to all those who have supported and contributed to the completion of this doctoral thesis. Without their assistance, and encouragement, this research would not have been possible.

First and foremost, I am immensely grateful to my supervisor, Dr. Driss Matrouf, for his invaluable guidance, and unwavering support. I am indebted to his mentorship and encouragement, which have shaped my research and nurtured my intellectual development.

I would like to extend my heartfelt thanks to the members of my thesis committee, Pr. Anthony Larcher, Dr. Pavel Matejka, Dr. Irina ILLINA, DR. Claude BARRAS, PR. Jean-Francois BONASTRE, DR. Mickael ROUVIER and PR. Fabrice LEFEVRE. Their insightful feedback, constructive criticism, and rigorous evaluation have significantly enriched the quality of this thesis. I am grateful to the faculty and staff of Avignon University, particularly the LIA (Laboratoire Informatique d'Avignon), for providing a stimulating academic environment.

Additionally, I would like to acknowledge the financial support provided by Agence nationale de la recherche (ANR) in the framework of Robovox project, which enabled me to pursue my research goals and attend conferences and workshops that enhanced my knowledge and academic network. Also, I appreciate the collaboration of Robovox members: Romain Serizel and Denis Jouvet from University of Lorraine, and Theophile GONOS from A.I.Mergence company.

I would like to express my deepest appreciation to my family for their love, encouragement, and belief in my abilities. I am deeply grateful to my wife, Mahrokh, for her sacrifices, patience, and words of wisdom, which have propelled me forward.

Last but not least, I extend my heartfelt appreciation to all my friends at Laboratoire Informatique d'Avignon for their encouragement, and moral support. Their presence has provided a much-needed balance during the demanding phases of this research.

Thank you all.

Mohammad MOHAMMADAMINI

CONTENTS

List of Figures

List of Tables

Acronyms

ANN	Artificial Neural Network
BT	Barlow Twins
CE	Cross Entropy
CNN	Convolutional Neural Network
DAE	Denoising Autoencoder
DCF	Decision Cost Function
DCT	Detection Cost Tradeoff
DET	Detection Error Trade-off
DFT	Discrete Fourier Transform
DNN	Deep Neural Networks
EER	Equal Error Rate
FA	False Acceptance
FAR	False Acceptance Rate
FFNN	Feedforward Neural Network
FFT	Fast Fourier Transform
FR	False Rejection
FRR	False Rejection Rate
GAN	Generative Adversarial Network
IBM	Ideal Binary Mask
IRM	Ideal Ratio Mask
LDA	Linear Discriminant Analysis
LR	Likelihood Ratio
MFCC	Mel Frequency Cepstral Coefficient
MHSA	Multi-Sead Self-Attention
MSE	Mean Square Error
PLDA	Probabilistic Linear Discriminant Analysis
ResNet	Residual Neural Network
SE block	Squeeze and Excitation block
TDNN	Time Delay Neural Network

ABSTRACT

Speaker recognition systems authenticate the identity of speakers from their speech utterances. In order to authenticate the identity of a claimed user, it is required to obtain a fixed-length compact speaker-discriminant representation for variable-length speech utterances known as speaker embeddings. The current speaker recognition systems are using DNNs to extract speaker embeddings. Despite the relative robustness of DNN-based speaker recognition systems, their performance degrades in the presence of acoustical variabilities such as additive noise and reverberation. There are three main groups of variabilities that reduce the performance of speaker recognition systems: internal (e.g. age, emotion, and stress), external (e.g. noise, reverberation, and distance), and content (e.g. language, and accent). The main theme of this thesis is robust DNN-based text-independent speaker recognition systems against additive noise and reverberation variabilities. The impact of variabilities can be addressed at the signal level, feature level, speaker embedding extractor, speaker embedding, and scoring adaptation techniques. The scope of our work is speaker embedding extractor and speaker embedding in two well-known and successful DNN-based speaker recognition systems: TDNN, and ResNet.

The first part of our work (Chapter 5) is on proposing several noise compensation DAEs (Stacked DAE, Gaussian DAE) that perform a transformation between pairs of distorted/clean speaker embeddings extracted by the TDNN system. The Stacked DAE is composed of several DAEs where each DAE receives as input the output of its predecessor DAE concatenated with the difference between noisy speaker embeddings and its predecessors' output. The noise compensation modules are tested in the case of additive noise (unseen noises, specific noise), early reverberation, and late reverberation distortions. We show a significant improvement of equal error rate in all cases ranging from 20% to 76% relative gain of equal error rate. In this part, we proposed two configurations for compensating multiple distortions.

In the second part of our work (Chapter 6), the behavior of the ResNet speaker recognition system against noise and reverberation was explored and compared with the TDNN system. Also, we investigate the noise compensation on ResNet speaker embeddings in two cases: 1) compensation of artificial noise with artificial data, and 2) compensation of real noise with artificial data. The second case is the most desired scenario because it

makes noise compensation affordable without having real data to train denoising techniques. The experimental results show that in the first scenario noise compensation gives significant improvement with TDNN while this improvement in ResNet is not significant. In the second scenario, we achieved a 15% improvement of EER over the VoiCes Eval challenge in both TDNN and ResNet systems. In most cases, the performance of ResNet without compensation is superior to TDNN with noise compensation.

In the next part (Chapter 7), we move towards learning noise-robust speaker embedding extractors. We propose two ResNet-based speaker recognition systems that make the speaker embedding more robust against additive noise and reverberation. The goal of the proposed systems is to extract speaker embeddings in noisy environments that are close to their corresponding speaker embedding in a clean environment. The first proposed system learns the same distribution for both noisy and clean environments. The second proposed system shifts the noisy speaker embeddings towards the distribution of the best-obtained system in a clean environment. In different situations with real and artificial noises and reverberation conditions, the modified systems outperform the baseline ResNet system. The proposed systems are tested with four evaluation protocols. In the presence of artificial noise and reverberation, we achieved a 19% improvement in EER. The main advantage of the proposed systems is their efficiency against real noise and reverberation. In the presence of real noise and reverberation, we achieved a 15% improvement in EER.

In the last part of our work (Chapter 8), we proposed a noise-robust self-supervised ResNet speaker recognition system based on the Barlow Twins loss function. The Barlow Twins objective function tries to optimize two criteria: First, it increases the similarity between two versions of the same signal (i.e. the clean and its augmented noisy version) to make the speaker embedding invariant to the acoustic noise. Second, it reduces the redundancy between the dimensions of the speaker embeddings which improves the overall quality of speaker embeddings. The experimental results on the Fabiole corpus show a 22 % relative gain in terms of EER in clean environments and an 18% improvement in the presence of noise with low SNR and reverberation.

ABSTRACT IN FRENCH

Les systèmes de reconnaissance du locuteur ont pour objectif d'authentifier des locuteurs à partir de leurs énoncés vocaux. Afin d'authentifier un utilisateur revendiqué, il est nécessaire d'obtenir une représentation de chaque énoncé, sous la forme d'un vecteur de taille fixe, contenant linformation permettant la séparation des locuteurs. Les systèmes de reconnaissance de locuteurs actuels utilisent des Réseaux de Neurones Profonds (RNP) pour extraire de telles représentations aussi appelées embeddings de locuteurs. Malgré la robustesse relative des systèmes de reconnaissance de locuteurs basés sur des RNP, leurs performances se dégradent en présence de variabilités acoustiques telles que du bruit additif et de la réverbération. Il existe trois principaux types de variabilités qui réduisent les performances des systèmes de reconnaissance du locuteur : interne (par exemple, l'émotion, l'âge, et le stress), externe (par exemple, le bruit, la réverbération, et la distance entre le locuteur et le microphone) et des variabilités liées au contenu (par exemple, la langue, et l'accent). Cette thèse se concentre sur la robustesse, face aux bruits additifs et aux réverbérations, des systèmes de reconnaissances du locuteur indépendante du texte, basés sur les RNP. L'impact de ces variabilités peut être traité au niveau du signal, au niveau des caractéristiques, de l'extracteur dembeddings du locuteur, des embeddings et des techniques d'adaptation. Notre travail porte principalement sur la robustesse de l'extracteur des embeddings et sur les embeddings pour deux types de systèmes bien connus : TDNN et ResNet.

La première partie de notre travail (Chapitre 5) consiste à proposer plusieurs Auto-Encodeurs de Débruitage (AED) (Pile d'AED, AED Gaussien) pour compenser le bruit au niveau des embeddings. Ces systèmes effectuent la transformation entre des embeddings, extraits avec le système TDNN, bruités et leur version propre. La Pile d'AED est composée de plusieurs AED où chaque AED reçoit en entrée la sortie de son prédécesseur concaténée avec la différence entre l'embedding du locuteur bruité et la sortie de ses prédécesseurs. Ces modules de compensation de bruit sont testés dans le cas de bruits additifs (bruits inconnues, bruits spécifiques), de distorsions de réverbération précoce et de réverbération tardive. Nos expériences montrent une amélioration significative du taux d'erreur égal (EER). Dans tous les cas, 20% à 76% de gain relatif de EER est obtenu. Dans cette partie, nous avons proposé deux configurations dans le cas d'avoir plusieurs distor-

sions acoustiques.

Dans la deuxième partie de cette thèse (Chapitre 6), le comportement des systèmes de reconnaissance de locuteur de type ResNet face au bruit et à la réverbération est étudié et comparé au système de type TDNN. Nous étudions également la compensation du bruit sur des embeddings extraits par ResNet dans deux cas : 1) la compensation d'un bruit artificiel avec des données artificielles et 2) la compensation d'un bruit réel avec des données artificielles. Le deuxième cas est le scénario le plus intéressant car il permet d'entraîner le système de débruitage sans disposer de données réelles. Les résultats expérimentaux montrent que dans le premier scénario, la compensation du bruit donne une amélioration significative pour des embeddings de type TDNN mais pas pour des embeddings de type ResNet. Dans le deuxième scénario, nous avons obtenu une amélioration de 15 % de l'EER sur le challenge VoiCes Eval pour les systèmes TDNN et ResNet. Dans la plupart des cas, les performances de ResNet sans compensation sont supérieures au TDNN avec compensation de bruit.

La partie suivante (Chapitre 7), se concentre sur l'apprentissage de systèmes d'extraction d'embeddings du locuteurs robustes au bruit. Nous proposons deux systèmes de reconnaissance du locuteur basés sur des ResNet qui rendent l'intégration du locuteur plus robuste contre le bruit additif et la réverbération. Le but des systèmes proposés est déviter la propagation du bruit du signal à l'embedding. De cette manière, les embeddings extraits dans des environnements bruités sont proches de leur version extraite dans un environnement non-bruité. Le premier système proposé apprend la même distribution pour les environnements bruyants et propres. La seconde propose un système qui impose aux embeddings de locuteurs pour environnement bruité de se déplacer vers la distribution du système le mieux obtenu dans l'environnement propre. Dans différentes situations avec des bruits réels et simulés et des conditions de réverbération, les systèmes modifiés surpassent le système ResNet de base. Les systèmes proposés sont testés avec quatre protocoles d'évaluation. En présence de bruit artificiel et de réverbération, nous avons obtenu une amélioration relatif de 19 % de l'EER. Le principal avantage des systèmes proposés est leur efficacité contre le bruit réel et la réverbération. En présence de bruit et de réverbération réels, nous avons obtenu une amélioration relatif de 15 % de l'EER.

Dans la dernière partie de notre travail (Chapitre 8), nous avons proposé un système de reconnaissance du locuteur, de type ResNet, auto-supervisé et robuste au bruit, basé sur la fonction de perte Barlow Twins. La fonction de coût de type Barlow Twins essaie d'optimiser deux critères. Premièrement, elle augmente la similarité entre deux versions du même signal (c'est-à-dire la version propre et sa version bruitée augmentée) pour rendre les embeddings invariants au bruit acoustique. Deuxièmement, elle réduit la redondance entre les dimensions des embeddings, ce qui améliore la qualité globale des embeddings

de locuteurs. Les résultats expérimentaux sur le corpus Fabiole montrent un gain relatif de 22% en termes d'EER dans des environnements propres et une amélioration de 18% en présence de bruit à faible SNR et réverbération.

ABSTRACT IN KURDISH

سیستمەکانی ناسینەوەی ئاخێوەر ناسنامەی کەسەکان لە ڕێگەی دەربڕینە دەنگییەکانەوە دەناسنەوە. ناسینەوەی ئاخێوەر لە ڕێگەی هەڵسەنگاندنی نواندنی پەستێوراوی ئەندازە-نەگۆڕی ئاخێوەر-دیارخەری دەنگەکان دەکرێت کە بە نواندنی ئاخێوەر دەناسرێت. سیستمە هەنووکەییەکانی ناسینەوەی ئاخێوەر بۆ گەیشتن بە نواندنی ئاخێوەر کەڵک لە تۆڕە دەماریی قوواڵەکان وەردەگرن. وێڕای خۆڕاگریی ڕێژەییان، هێشتا سیستمەکانی ناسینەوەی ئاخێوەری دامەزراو لەسەر بنەمای تۆڕە دەماریی قوواڵەکان لەهەمبەر بگۆڕە ئاکۆستیکییەکان وەک نۆیزی زێدە و زایەڵە کارایی دادەبەزێت. سێ گرووپی سەرەکی لە بگۆڕەکان هەن کە کارایی سیستمەکانی ناسینەوەی ئاخێوەر دادەبەزێنن: ناوەکی (وەک ڕەگەز، تەمەن و سترێسی ئاخێوەر)، دەرەکی (وەک ژاوەژاو، زایەڵە و مەودای مایکرۆفۆن)، و ناوەرۆک (وەک زمان و بێژە). تێمی سەرەکیی ئەم تێزە ناسینەوەی ئاخێوەری خۆڕاگری دامەزراو لەسەر بنەمای تۆڕە دەماریی قوواڵەکان لەهەمبەر نۆیزی زێدە و زایەڵەیە. چارەسەری کێشەی بگۆڕەکان دەکرێت لە ئاستەکانی سیگناڵ، تایبەتمەندییە ئاکۆستیکییەکان، دەرهێنەری نواندنی ئاخێوەر، ئاڵوگۆڕی نواندنی ئاخێوەر یان تەکنیکەکانی پلەبەخشیدا بکرێت. ئێمە لەسەر دوو ئاستی دەرهێنەری نواندنی ئاخێوەر و ئاڵوگۆڕی نواندنی ئاخێوەر لە دوو سیستمی TDNN و ResNetدا کار دەکەین.

بەشی یەکەمی کارەکان پێشنیارکردنی چەند دژەنۆیزە (Stacked DAE, Gaussian DAE) کە بە ئاڵوگۆڕێک لەسەر جووتەی نۆیزی-خاوێن لە نواندنی ئاخێوەران هەوڵ دەدات کاریگەریی بگۆڕەکان لەسەر نواندنی ئاخێوەری دەرهێنراو لە TDNN کەم بکاتەوە. دژەنۆیزی Stacked DAE لە سەریەکنانی چەندین DAE پێکدێت کە هەر DAEیەک دەرچووی DAEی پێشووی لەگەڵ جیاوازیی نێوان نواندنی ئاخێوەری نۆیزی و دەرچووی DAEی پێشووی بە یەکەوە وەردەگرێت. سەرجەم دژەنۆیزەکان بۆ لابردنی نۆیزی زێدە (نۆیزی نەبینراو، نۆیزی دیاریکراو)، پێش زایەڵە، و پاش زایەڵە بە کار هێنراون. پیشانی دەدەین کە لە هەموو دۆخەکاندا ڕێژەی هەڵەی یەکسان (EER) لە 20 لەسەد تاکوو 76 لەسەد کەم دەبێتەوە. لەم بەشەدا دوو چوارچێوەمان بۆ ئەو حاڵەتانە پێشنیار کردوون کە زیاتر لە یەک بگۆڕی تێکدەر هەیە.

لە بەشی دووەمی کارەکەماندا، لە هەڵسوکەوتی سیستمی ناسینەوەی ئاخێوەری ResNet لەهەمبەر نۆیز و زایەڵەدا دەکۆڵینەوە و بە سیستمی TDNN بەراوردی دەکەین. هەروەها لابردنی نۆیز لەسەر نواندنی ئاخێوەری ResNet لە دوو

سیناریۆدا تاقی دەکەینەوە: 1. لابردنی نۆیزی دەستکرد لە ڕێگەی دراوەی دەستکرد 2. لابردنی نۆیزی ڕاستەقینە لە ڕێگەی دراوەی دەستکردەوە. حاڵەتی دووەم زۆر پڕبایەخە چونکە بەبێ هەبوونی دراوەی ڕاستەقینە و ئاگاداری لە ژینگە، لابردنی نۆیز کردەیی دەکات. ئەنجامی تاقیکردنەوەکانمان پیشانی دەدات کە لە حاڵەتی لابردنی نۆیزی دەستکرد لە سیستمی TDNN بە ئەنجامێکی سەرنجڕاکێش دەگەین، لە کاتێکدا دەستکەوتەکان بۆ سیستمی ResNet زۆر سنووردارن. لە سیناریۆی دووەمدا بۆ هەر دوو سیستمی TDNN و ResNet ڕێژەی هەڵەی یەکسان 15 لەسەد لەسەر کۆمەڵە دراوەی VoiCes کەم دەبێتەوە. هەروەها ئەنجامەکان پێمان دەڵێن کە لە زۆربەی حاڵەتەکاندا سیستمی ResNet بەبێ لابردنی نۆیزیش لە سیستمی TDNN خۆڕاگرترە.

لە بەشی دواتردا بەرەو ڕاهێنانی خۆڕاگری دژەنۆیزی سیستمەکانی ناسینەوەی ئاخێور دەچین. دوو سیستمی ناسینەوەی ئاخێوەر لەسەر بنەمای ResNet بۆ ئەم مەبەستە پێشنیار دەکەین کە هەوڵ دەدەن نواندنی ئاخێوەری خۆڕاگر لە هەمبەر نۆیز و زایەڵەدا بەرهەم بهێنن. ئامانجی سیستمە پێشنیارکراوەکان ئەوەیە کە تا ئەو جێگەی دەکرێت نواندنی ئاخێوەری یەکسان بۆ دوو ژینگەی بێنۆیز و نۆیزی دەربهێنرێت. یەکەم سیستمی پێشنیارکراو دابەشێنێکی یەکسان بۆ هەردوو ژینگەی نۆیزی و بێنۆیز فێر دەبێت. لە دووەم سیستمی پێشنیارکراودا ئەوە بەسەر سیستمەکەدا دەسەپێنین کە تا ئەو شوێنەی دەکرێت بەرەو دابەشینی باشترین ژینگەی بێنۆیز بڕوات. سیستمە پێشنیارکراوەکان لەسەر چوار پرۆتۆکۆڵی هەڵسەنگاندن تاقی کراونەتەوە. لە حاڵەتی هەبوونی نۆیز و زایەڵەی دەستکرددا، ڕێژەی هەڵەی یەکسان 19 لەسەد کەم دەبێتەوە. لایەنی باشی ئەم سیستمانە ئەوەیە کە لە بەرامبەر نۆیز و زایەڵەی ڕاستەقینەشدا کاران. لە حاڵەتی هەبوونی نۆیز و زایەڵەی ڕاستەقینەدا ڕێژەی هەڵەی یەکسان 15 لە سەد کەم دەکەنەوە.

لە دواییین بەشی توێژینەوەکاندا، سیستمێکی ناسینەوەی ئاخێوەری خۆچاودێریکەری پتەو لە بەرامبەر نۆیزدا پێشنیار دەکەین کە کەڵک لە نەخشەی ئامانجی بارلۆ تویننز وەردەگرێت. نەخشەی ئامانجی بارلۆ تویننز هەوڵ دەدات دوو پێوەر باشینە بکات: یەکەم، هاوشێوەیی نێوان دوو وەشانی سیگنالّ (لێرەدا نۆیزی و خاوێن) زیاد بکات، تاکوو نواندنی ئاخێوەر لەبەرامبەر نۆیزدا نەگۆڕ دەکات. دووەم، زانیاریی فرەپاتە لەنێوان ڕەهەندەکانی نواندنی ئاخێوەردا کەم دەکاتەوە، تاکوو کارایی گشتیی سیستمەکە لە هەموو ژینگەیەکدا ببات سەر. تاقیکردنەوەکان پیشانی دەدەن کە لەسەر کۆرپسی Faboile ڕێژەی هەڵەی یەکسان بۆ ژینگەی بێنۆیز 22 لەسەد و لە حاڵەتی هەبوونی نۆیز و زایەڵە بە ڕێژەی نزمی سیگنالّ لە هەمبەر نۆیزدا، کارایی سیستم 18 لەسەد باشتر دەبێت.

1

INTRODUCTION TO SPEAKER RECOGNITION SYSTEMS

Lacking any data lowers its threshold for noise, which then gets amplified into arbitrary patterns of signals, producing, eventually, detailed hallucinations.

Daniel Dennett

In this introductory chapter, the general framework of speaker recognition systems is presented. After that, a taxonomy of speaker recognition systems is given, in which we postulate the domain of the considered systems in this thesis as text-independent open-set speaker identification systems. In the next section, the challenges affecting the performance of speaker recognition systems are reviewed. Since the domain of this thesis is the robustness of the speaker recognition systems in the presence of additive noise and reverberation, the negative impact of these acoustical variabilities on speech signal and speaker recognition systems is discussed in more detail. This thesis is done in the framework of the RoboVox project, a mobile robot equipped with a speaker recognition system. The challenges facing a speaker recognition system, in this case, are discussed in this chapter.

1.1. INTRODUCTION

SPEAKER recognition systems are among the well-suited and well-known applications in the domain of speech processing. Generally speaking, the goal of a speaker recognition system is the authentication of speakers' identities from their speech utterances. Speaker recognition systems have introduced new methodologies and challenges into the domain of speech processing, however, they have commonalities with other speech technologies. The last generation of speaker recognition systems is mainly based on deep neural networks [1–5]. Acoustical variabilities such as far-field speech, additive noise, and reverberation are among the historical challenges that degrade the performance of speaker recognition systems [6–9]. Although DNN-based systems are more plausible in facing acoustical variabilities, still they are suffering from these challenges.

Speaker recognition systems are ubiquitous. They can be used as a separate system or they can be integrated into other speech applications such as automatic speech recognition systems [10]. The broad range of speaker recognition applications includes voice indexing and voice search, teleconferencing, finance, access control, surveillance, and forensic and legal applications [11]. This broad usage of speaker recognition systems exposes them to adversarial environments with different kinds of variabilities. The main topic of this thesis is to address the problem of additive noise and reverberation distortions in the domain of DNN-based speaker recognition systems.

In the discussion of DNN-based speaker recognition systems, it is convenient to separate the training and application phases. In the same manner, it is needed to discuss the datasets in training and application parts separately. Our discussions here are concise and different modules are scrutinized in the next chapters more precisely.

In the training phase, a deep neural network classifies the acoustic features based on the speaker label. The implication of speaker classification results in producing a speaker discriminant representation at the utterance level obtained from deep hidden layers [1, 2]. This representation is known as speaker embedding, speaker representation, and x-vector. The x-vector is adopted from the first successful DNN-based speaker recognition systems and we just use this term for TDNN-based speaker embeddings [2]. For other DNN-based systems, we use speaker embedding throughout the thesis.

In the application phase, the trained speaker embedding network is used to generate a fixed-length representation for speech utterances that contains speaker discriminant features. The speaker classification part will be removed and the fixed speaker embedding extractor will be held. In this step, we need enrollment and test datasets.

The enrollment dataset includes files for the registered or new speakers whose identities should be verified/recognized. The test data belongs to speakers who claim the system belongs to them. The trained speaker embedding extractor gives speaker discriminant representations with the same length for comparison. The comparison will be done by a scoring technique: if the similarity between test and enrollment speaker embeddings is higher than a threshold, the user will be accepted; otherwise, it will be rejected. The training and application steps are depicted in Figure 1.1.

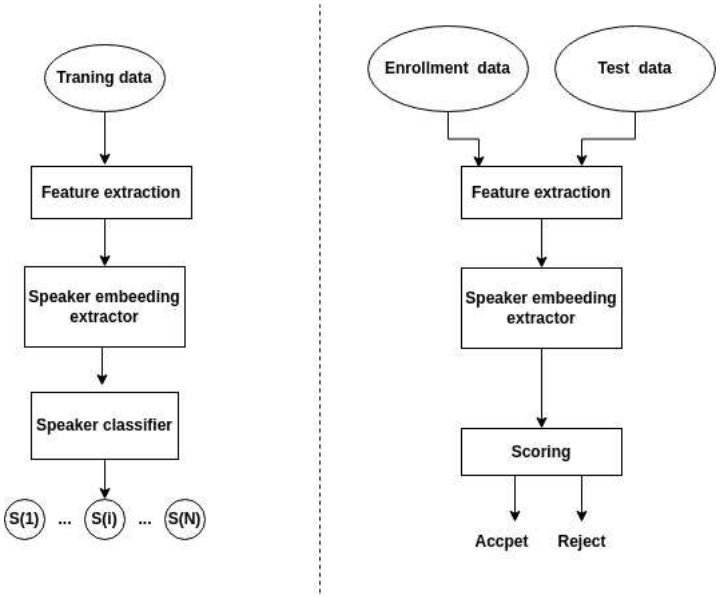


Figure 1.1: Configuration of a DNN-based speaker recognition system. *Left:* Traning phase *Right:* Application phase

1.2. A TYPOLOGY OF SPEAKER RECOGNITION SYSTEMS

The speaker recognition systems can be categorized in terms of the task, the content of the spoken dialog, and speakers:

- ***text-independent vs. text-dependent:*** Text-dependent speaker recognition systems expect specific utterances while text-independent speaker recognition systems are not constrained in terms of the content pronounced by the users.
- ***open-set vs. close-set:*** In close-set systems the list of clients are fixed but in the open-set configuration adding new clients is allowed. In an open-set configuration when the similarity between a claimed user and the registered clients is less than a threshold, it can be considered as a new client.

- ***identification vs. verification:*** Speaker verification is a one-to-one comparison in order to determine if the voice of the claimed user comes from a specific speaker. But in speaker identification, the comparison is one-to-many, the claimed user is compared with a pool of registered users.

The major part of our work in this thesis falls into the text-independent open-set speaker identification setup (Figure. 1.2). It is worth mentioning that the type of the system can bring some variabilities and make speaker recognition more difficult. For example, text-independent is more challenging than text-dependent speaker recognition. Other types of variabilities that can impact the performance of speaker recognition systems are presented in the next section.

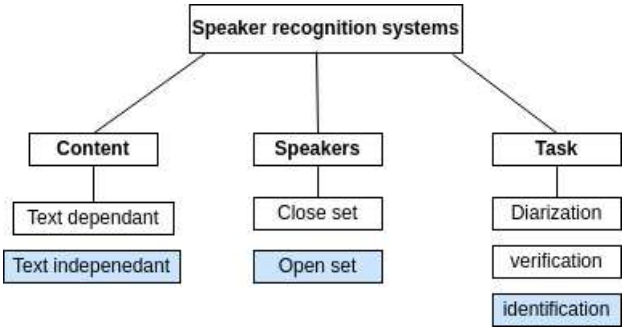


Figure 1.2: A typology of different speaker recognition systems based on the content of the spoken dialog, the possibility of adding new speakers, and the number of system owners.

1.3. SPEAKER RECOGNITION VARIABILITIES AND CHALLENGES

There are different types of variabilities that impact the performance of speaker recognition systems. A general categorization of these variabilities is shown in Figure 1.3. Three main categories are internal variabilities, external variabilities, and content-based variabilities. The highlighted variabilities (i.e. additive noise, reverberation, far-field speech) are the three addressed challenges in this thesis.

1.3.1. INTERNAL VARIABILITIES

There are many reasons that impact the way of speaking for a specific person. A given speaker even doesn't pronounce the same content in different situations. This alteration of speaking is called session variability, within-speaker variability, or internal variability. Different kinds of internal variabilities are:

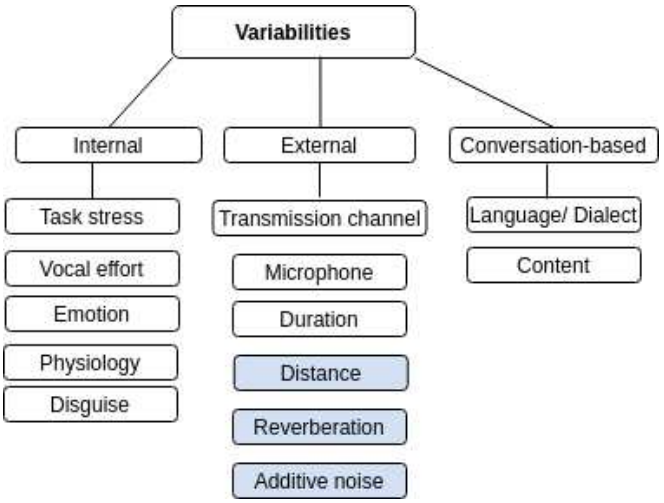


Figure 1.3: Different kinds of variablities

- **Task stress.** Uttering speech signals during a stressful task such as driving will be impacted significantly. Speaking under stress can impact the speed of speaking or even slurred speech. The evaluation of speaker recognition systems performance shows the significant impact of speaking under stress. Making the system robust in such situations has been addressed in several papers [12, 13].
- **Vocal effort.** When a severe background noise becomes a barrier to communication, the speaker changes his manner of speaking. The Lombard effect is a well-known vocal effort that refers to the speaker being louder involuntarily in the presence of noise. The speech signal under the Lombard effect is different from a normal speech in several ways, including increased intensity, pitch, glottal spectral tilt, etc. It is shown that the Lombard effect impacts the performance of speaker recognition systems [14].
- **Emotion.** It has been shown that speaker embedding contains information about emotions. This information impacts the performance of speaker recognition systems significantly [15].
- **Physiological.** The speech disorders and changing voice over age are examples of physiological variabilities that can cause a domain mismatch in speaker recognition systems[16].
- **Disguise.** The deliberate or non-deliberate modification of speech such as impersonation or mimicking of speaking a foreign language that requires more effort than normal speech is called disguising [17].

1.3.2. EXTERNAL VARIABILITIES

External variabilities arise from the acoustical environment, recording devices, and transmission channels. This group of variabilities is not concerned about "who" is speaking and "what" is pronounced. The external variabilities are so prevalent that making the speaker recognition system robust against them embraces a major body of speaker recognition research. The main external variabilities are:

- **Transmission channel and microphone.** The sender and receiver's user interface and the bandwidth limitation degrade the quality of speech signal that can impact the performance of the speaker recognition systems [18].
- **Additive noise.** All sounds except the speaker's utterances in the environment are called additive noise. Among common additive noises, we can mention water sounds, electrical device sounds, animal sounds, babble noises, etc.
- **Reverberation.** Reverberation is the impact of sound reflection on the speech signal. This repetitive reflection of sound over the surfaces reduces the performance of speaker recognition systems severely. Additive noise and reverberation are discussed in the next section in more detail.
- **Duration.** The speaker recognition systems perform well in the presence of enough data [19]. The lack of information in short speech signals reduces the performance of the speaker recognition systems severely. In some cases, having enough data is impossible and it is required to increase the robustness of the system for such situations.
- **Distance.** The distance between the speaker and microphone impacts the quality of the speech signal and reduces the SNR. In the far-distance speech, the attenuation of speech signal causes a severe impact of additive noise and reverberation [20].

1.3.3. CONVERSATION

- **Language or dialect**. DNN-based speaker embeddings carry language and dialect information [21]. Language mismatch can arise in different steps of the speaker recognition process such as a mismatch between the training or application data or a language mismatch between test and enrollment utterances. [22].
- **Content**. The speaker embeddings carry the content information [23]. In the same acoustical situation, the speaker embeddings for the same or similar utterances are closer in comparison to utterances with completely different content. Therefore, the content mismatch between target and test utterances

impacts speaker recognition detrimentally.

1.4. ENVIRONMENTAL VARIABILITIES

Among the above-mentioned variabilities, some of them such as environmental variabilities are more prevalent and have more negative effects. Additive noise and reverberation are the main environmental variabilities. When there is more than one variability their negative impact becomes more serious. For example, the presence of additive noise and reverberation with a short duration makes the system even more infeasible [20].

In order to reduce the impact of variabilities, two general approaches can be taken. The first approach tries to improve the general performance of the system in all environments, while the second approach tries to target a specific variability. For example, the DNN-based approaches are generally more robust facing different internal or external variabilities in comparison to their precedent statistical methods such as i-vector systems, but still, they need to be made more robust. In this thesis, we try to take both approaches but the main focus is on the second one. The additive noise and reverberation are the two variabilities that have been considered throughout the thesis. In the next subsections, the impact of additive noise and reverberation on speech signals is discussed deeply.

1.4.1. ADDITIVE NOISE

Environmental noise is omnipresent. In different environments such as streets, restaurants, libraries (air conditioning), and cars (engines), where speaker recognition systems are used, we are facing background noises. Those noises that stay during the time are called stationary noises, while those that are changing over time are called non-stationary noises. In the case of stationary noise, because the nature of the noise is known, the noise information can be used in different noise suppression approaches. While in the case of non-stationary noise, the nature of the noise is unknown and can be changed over time. This feature makes noise formulation more challenging. A robust speaker recognition system should handle both stationary and non-stationary noises. Figure 1.4 shows the impact of different variabilities including additive noise.

In Equation 1.4, the general formulation of variabilities including additive noise is given:

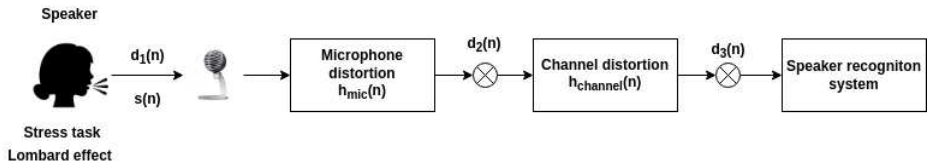


Figure 1.4: Modeling the impact of different variabilities on speech signal [24]

$$y(n) = \Big(\Big(\Big\{\Big[s(n)\Big|Task\ stress,\ Lombard\ effect\Big] + d_1(n)\Big\} \otimes h_{mic}(n) + d_2(n)\Big) \otimes h_{channel}(n)\Big) \tag{1.1}$$

where:

- $s(n)$ is noise-free speech signal
- $d_1(n)$ is additive noise
- $h_{mic}(n)$ is microphone mismatch distortion
- $h_{channel}(n)$ is convolutive noise of a transmission channel
- $d_2(n)$ is additive noise caused by transmission channel
- $d_3(n)$ is background noise from the side of the receiver
- $y(n)$ is degraded speech signal

In the case of additive noise, just the $d_1(n)$ will be considered.

The signal-to-noise ratio (SNR) is a common metric to measure the intensity of noise. The SNR is defined as:

$$SNR = 10log_{10}(\frac{P_{Signal}}{P_{noise}}) \tag{1.2}$$

where P_{signal} and P_{noise} are the power of signal and noise respectively. For high values of SNR, the impact of noise on speaker recognition systems is not significant.

1.4.2. REVERBERATION

When the speaker is in a closed space and far from the recording devices, the speech signal arrives from different paths to the microphone. However the speech signal can directly move from the mouth of the speaker to the recording device, it can also

be reflected from different surfaces and objects. The reflected speech wave is called reverberation which impacts the acoustic characteristics of the speech signal. Indeed, the reverberations are attenuated copies of the original speech signal.

The impact of reverberation can be modeled as a linear convolution of the speech signal and room impulse response as Equation 1.3:

$$y(t) = \Sigma_{\tau}^{T} r(\tau) x(t-\tau) = r(t) \otimes x(t) \tag{1.3}$$

where $y(t)$, $r(t)$, and $x(t)$ represent the reverberated signal, room impulses, and clean signal respectively.

The room impulse response describes the changes of speech signal reflected from the surface. They depend on many factors such as:

- ***Angle:*** The angle between the speaker and microphone changes the path between the speaker and microphone. The indirect path causes more distortion.
- ***Distance:*** The far-distance speech causes more attenuation of the speech signal.
- ***Absorption rate:*** The absorption rate of speech signal for different surfaces is different [25].
- ***Size and shape of room.*** In the big closed spaces the impact of reverberation increases. For L-Shape [6] or curved rooms where the path between the speaker and the microphone is not direct the impact of reverberation increases.
- ***Position of the microphone.*** When the height of the microphone and speaker are not the same.

For a sinewave-modulated signal shown as:

$$I(t) = I_{INP}.(1 + cos(2\pi F t) \tag{1.4}$$

where F is Frequency(HZ), the reverberated sinewave-modulated signal shown as

$$I(t) = I_{INP}.(1 + m.cos(2\pi F t) \tag{1.5}$$

where m is called the modulation depth or modulation. The smaller value of m means more reverberation. This change in modulation caused by reverberation is called smearing. This phenomenon for different values of m is shown in Figure 1.5.

It is important to differentiate between two types of reverberations: early reverberation and late (full) reverberation. After the arrival of direct speech, several strong reflections called early reverberation will arrive. The early reverberation occurs $50ms$ after the arrival of the direct signal. After that, numerous indistinguishable reflections arrive that are called late reverberation [26]. The required time for the late

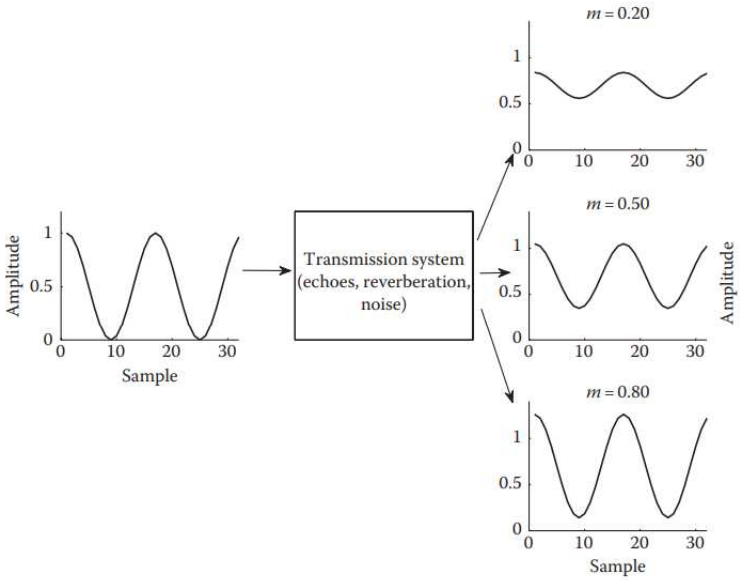


Figure 1.5: The smearing of the speech signal by reverberation[26]

reverberation to decay by 60 dB relative to the level of the direct sound is called the reverberation time T_{60}. For a normal room the T_{60} range between 200ms and 1000ms. The early reverberations are strongly dependent on the speaker and microphone position. The difference between early and late reverberation is shown in figure 1.6.

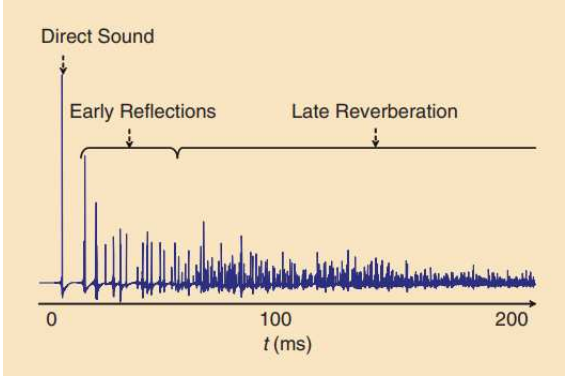


Figure 1.6: The differences between direct speech, early reverberation, and late reverberation [26]

In many real scenarios where speaker recognition systems are used, both

reverberation and additive noise distortions are present. The presence of both additive noise and reverberation can be modeled as:

$$y(t) = \Sigma_{\tau}^{T} r(\tau) x(t-\tau) + d(t) = r(t) \otimes x(t) + d(t) \tag{1.6}$$

where $r(t)$ stands for room impulses and $d(t)$ stands for additive noise.

The impact of additive noise and reverberation is strongly dependent on the distance between the microphone and the speaker. If we compare the impact of additive noise and reverberation in the same environment for two microphones, a closed microphone and a far one, we observe the severe attenuation of the speech signal in the case of a far microphone that leads to very low SNR. As an example, Figure 1.7 shows the same utterance recorded in a closed room simultaneously with a closed microphone compared to the far microphone. The close microphone is near the mouth and the far microphone is located $3m$ far from the speaker.

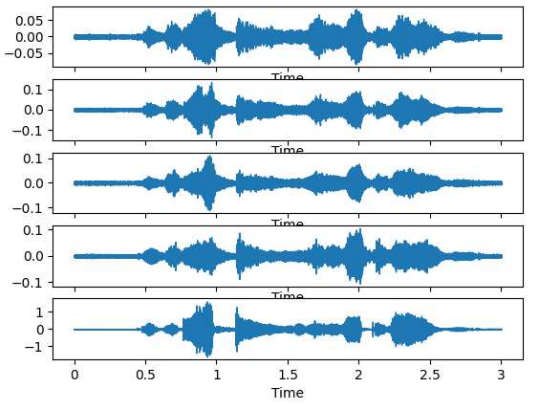


Figure 1.7: The impact of noise and reverberation in the far (first four signals) and close (the last signal) microphone (adopted from RoboVox dataset)

1.5. ROBOVOX PROJECT

This thesis is done in the framework of the RoboVox project. Robovox is a mobile security robot that is equipped with a speaker recognition system. The embedded speaker recognition system faces several challenges relating to the remote identification of a person in real conditions, which can reduce its performance drastically[1]:

[1] https://robovox.univ-avignon.fr/

- **Ambient noise leading to low signal-to-noise ratios (SNR):** The speech signal is distorted with noise from fans, air conditioners, heaters, computers, etc.
- **Internal robot noises (robot activators):** The robot's activator noise reverberates on the audio sensors and degrades the SNR.
- **Reverberation**: The phenomena of reverberation due to the configuration of the places where the robot is located. The robot is used in different rooms with different surface textures and different room shapes and sizes.
- **Distance:** The distance between the robot and speakers is not fixed and it is possible for the robot to move during the recognition.
- **Babble noise:** The potential presence of several speakers speaking simultaneously.

1.6. ROAD-MAP OF THE THESIS

- ***Chapter 2.*** In this chapter different modules of DNN-based speaker recognition systems are described. Among the speaker embedding networks TDNN, ECAPA-TDNN, MFA-Conformer, and ResNet networks are reviewed.
- ***Chapter 3.*** In this chapter, the robust speaker recognition system approaches are reviewed. Among the main approaches speech enhancement techniques, robust speaker embeddings, and noise compensation at the speaker embedding level are discussed.
- ***Chapter 4.*** In this chapter, the known datasets for training speaker embedding networks, data augmentation, and speaker recognition evaluation are discussed.
- ***Chapter 5.*** In this chapter, a general framework for noise compensation is proposed that is based on mapping noisy x-vectors to their clean version. Several systems are proposed to do this transformation. Also, the compensation for the joint presence of noise and reverberation is explored.
- ***Chapter 6.*** In this chapter, noise compensation is explored in ResNet and TDNN systems. The limitation of noise compensation in speaker embedding level is studied.
- ***Chapter 7.*** In this chapter, two variants of ResNet speaker embedding extractors have been proposed that shift the noisy embeddings towards their clean corresponding distribution.
- ***Chapter 8.*** A self-supervised framework proposed that reduces the redundancy between dimensions of speaker embeddings and makes them invariant to noise and reverberation.

1.7. PERSONAL BIBLIOGRAPHY

During the preparation of this thesis 7 national and international conference papers are published:

[1] **Mohammadamini, M.**, Matrouf, D., Noé, P.-G. (2020) *Denoising x-vectors for Robust Speaker Recognition.* Proc. The Speaker and Language Recognition Workshop (Odyssey 2020), 75-80, doi: 10.21437/Odyssey.2020-11

[2] **M. Mohammad Amini** and D. Matrouf, *Data augmentation versus noise compensation for x-vector speaker recognition systems in noisy environments*, 2020 28th European Signal Processing Conference (EUSIPCO), Amsterdam, Netherlands, 2021, pp. 1-5, doi: 10.23919/Eusipco47968.2020.9287690

[3] **M. Mohammadamini**, D. Matrouf, J. -F. Bonastre, R. Serizel, S. Dowerah and D. Jouvet, *Compensate multiple distortions for speaker recognition systems*, 2021 29th European Signal Processing Conference (EUSIPCO), Dublin, Ireland, 2021, pp. 141-145, doi: 10.23919/EUSIPCO54536.2021.9615983.

[4] **Mohammadamini, M.**, Matrouf, D., Dowerah, S., Serizel, R., Jouvet, D., Bonastre, J.-F. (2022) *Le comportement des systèmes de reconnaissance du locuteur de l'état de l'art face aux variabilités acoustiques.* Proc. XXXIVe Journées d'Études sur la Parole – JEP 2022, 241-249, doi: 10.21437/JEP.2022-26

[5] **M. MohammadAmini**, D. Matrouf, J. -F. Bonatsre, S. Dowerah, R. Serizel and D. Jouvet, *A Comprehensive Exploration of Noise Robustness and Noise Compensation in ResNet and TDNN-based Speaker Recognition Systems*, 2022 30th European Signal Processing Conference (EUSIPCO), Belgrade, Serbia, 2022, pp. 364-368, doi: 10.23919/EUSIPCO55093.2022.9909726.

[6] **MohammadAmini, M.**, Matrouf, D., Bonastre, J.-F., Dowerah, S., Serizel, R., Jouvet, D. (2022) *Learning Noise Robust ResNet-Based Speaker Embedding for Speaker Recognition.* Proc. The Speaker and Language Recognition Workshop (Odyssey 2022), 41-46, doi: 10.21437/Odyssey.2022-6

[7] **Mohammadamini, M.**, Matrouf, D., Bonastre, J.-F., Dowerah, S., Serizel, R., Jouvet, D. (2022) *Barlow Twins self-supervised learning for robust speaker recognition.* Proc. Interspeech 2022, 4033-4037, doi: 10.21437/Interspeech.2022-11301

Also, the following papers are published in the same domain that are closely related to the topic of the thesis:

[8] Noé, P.-G., **Mohammadamini, M.**, Matrouf, D., Parcollet, T., Nautsch, A., Bonastre, J.-F. (2021) Adversarial Disentanglement of Speaker Representation for Attribute-Driven Privacy Preservation. Proc. Interspeech 2021, 1902-1906, doi:

10.21437/Interspeech.2021-1712

[9] Sandipana Dowerah, Romain Serizel, Denis Jouvet, **Mohammad Mohammadamini,** Driss Matrouf. *How to Leverage DNN-based speech enhancement for multi-channel speaker verification?*. 4th International Conference on Advances in Signal Processing and Artificial Intelligence (ASPAI 2022), Oct 2022, Corfu, Greece.

[10] S. Dowerah, R. Serizel, D. Jouvet, **M. Mohammadamini** and D. Matrouf, *Joint Optimization of Diffusion Probabilistic-Based Multichannel Speech Enhancement with Far-Field Speaker Verification*," 2022 IEEE Spoken Language Technology Workshop (SLT), Doha, Qatar, 2023, pp. 428-435, doi: 10.1109/SLT54892.2023.10022350.

[11] Mickael Rouvier and **Mohammad Mohammadamini**. 2022. *Far-Field Speaker Recognition Benchmark Derived From The DiPCo Corpus*. In Proceedings of the Thirteenth Language Resources and Evaluation Conference, pages 19551959, Marseille, France. European Language Resources Association.

[12] Amani, A., **Mohammadamini, M.**, Veisi, H. (2021). *Kurdish Spoken Dialect Recognition Using X-Vector Speaker Embedding*. In: Karpov, A., Potapova, R. (eds) Speech and Computer. SPECOM 2021. Lecture Notes in Computer Science(), vol 12997. Springer, Cham. https://doi.org/10.1007/978-3-030-87802-3$_5$

2

DNN-BASED SPEAKER RECOGNITION SYSTEMS

We understand what it means for a device to respond to a red light or a loud noise but humans are the only devices in the universe that respond to danger.

Steven Pinker

In this chapter, we provide an overview of DNN-based speaker recognition systems. Our discussion will include feature extraction, speaker embedding extractors, and speaker recognition back-ends. We begin with classical feature extraction methods including MFCCs and Filter banks. After that, the unsupervised DNN-based speech representation methods including Wav2Vec and WavLM are described. The core of DNN-based speaker recognition systems is the speaker embedding extractor. In this chapter, we show the timeline of evolving speaker embedding extractors. Among them, the d-vector, TDNN, ResNet, ECAPA-TDNN, and MFA-Conformer systems are described. We will describe how each system improved its predecessor. The last part of this chapter is devoted to speaker recognition back-ends. The common PLDA and Cosine scoring techniques are reviewed. Finally, the well-known speaker recognition metrics including EER, DCF, and DET are presented.

2.1. INTRODUCTION

Deep learning methods have revolutionized all aspects of language and speech processing. The speaker recognition systems are not exempt from this revolution. The current speaker recognition systems are based on deep neural networks. In the previous chapter, the overall process of a DNN-based speaker recognition system was described from both training and application perspectives. In this chapter, we delve into all parts of these systems in more detail.

First of all, the feature extraction steps are described. The goal of feature extraction is to produce compact and less redundant characters of speech signals. We begin with classical feature extraction methods. The MFCC and filter banks are described which are the main features used in the contribution part. Deep learning methods are revolutionizing feature extraction methods too. We will describe the new DNN-based speech representation methods such as WavLM and Wav2Vec in Section 2.2.3.

Speaker modeling is the process of obtaining a compact fixed-length speaker discriminant representation from a variable-length speech utterance. The main reason behind the success of current speaker recognition systems is the powerful ability to learn speaker representations. The DNN-based speaker embedding extractors are powerful to extract speaker discriminant characteristics from the speech signals. Whether these characteristics are known such as gender, age, and accent or they are unknown speaker features. The second part of this chapter is devoted to DNN-based speaker modeling methods that are used to extract fixed-length speaker embeddings. In Section 2.3 the evolving process of DNN-based speaker modeling is described. After a short revision of statistical speaker modeling methods, the main DNN architectures used for speaker modeling are described (Section 2.4).

A DNN-based speaker modeling framework has three main parts: Frame-level DNN architecture, pooling layer, and speaker classification. Among the DNN architectures, we will discuss the systems based on TDNN, ResNet, ECAPA-TDNN, and MFA-Conformer. The common pooling methods including average pooling, statistics pooling, and attentive pooling are described. The speaker classifier optimizes a multi-class objective function. The softmax and angular softmax are among the revised objective functions that are used during the training of the speaker embedding networks.

After removing the speaker classifier part from the DNN, the remaining part is used as a speaker embedding extractor. The speaker embeddings extracted by means of this network are used for scoring. Dimensionality reduction is an important preprocessing step before scoring, which is discussed in this chapter. In the last part of this chapter, the Cosine, and PLDA scoring techniques are discussed. Finally, we will discuss

various evaluation metrics used in speaker recognition systems.

2.2. FEATURE EXTRACTION

2.2.1. VOICE ACTIVITY DETECTION

In each speech signal, there are some silent parts that don't have speaker discriminant information and introduce unwanted information into the speaker embeddings. Detecting the silent parts of a speech signal is called Voice Activity Detection (VAD) which is a crucial prepossessing step in speaker recognition systems. There are several VAD algorithms. The simplest and most widely used algorithm is based on energy threshold. The energy for $s(t)$, a speech signal between t_1 and t_2, is defined as.

$$E = \int_{t=t_1}^{t_2} x^2(t)\, dx \tag{2.1}$$

In the energy-based VAD, a speech frame is considered as silent if the energy is less than a threshold. This threshold can be variable according to the speaker, background noise, etc. The precision of VAD impacts the performance of speaker recognition systems significantly [27]. Zero-crossing VAD is another well-known method. The zero-crossing number for $s(t)$ between t_1 and t_2 is the number of points where $s(t) = 0$. There are more sophisticated methods such as statistical model (SM) VAD, gaussian mixture model (GMM) VAD [27], and DNN-based VAD [28]. Throughout this thesis, an energy-based VAD implemented in the Kaldi toolkit is used [29].

2.2.2. FILTER BANK AND MFCC

Feature extraction is a key component of speaker recognition systems. The aim is to create a compact, less-redundant representation of speech signals that can be used more easily than raw signals by DNNs. Filter banks and MFCCs are among the common features used to train DNN-based speaker recognition systems. The extraction steps of these features are as follows [30]:

1. **Framing:** Framing is the speech signal segmentation into very short parts. Normally, the length of each frame is 25 milliseconds and each frame has an overlap with its context frames.
2. **Pre-emphasis:** Since in high-frequency speech signals the magnitude is less than low frequency, the pre-emphasis operation is done in order to increase the magnitude of the speech signal in high frequency. For a speech signal, $s[n]$, the

pre-emphasis is defined as:

$$y[n] = s[n] - As[n-1] \tag{2.2}$$

where A is between 0.9 and 1.

3. **Windowing:** In order to reduce the impact of spectral leakage at the beginning and end of a frame, a windowing technique is applied to each frame. The hamming window is a common windowing function that is defined as:

$$w[n] = 0.54 - 0.46\cos(\frac{2\pi n}{N-1}), n = 0, 1, ..., N-1 \tag{2.3}$$

where N is the frame size.

4. **FFT:** The Discrete Fourier Transform (DFT) brings the speech signal from the time domain into the frequency domain. Because in the frequency domain, the speech characteristics such as formants and pitches are better revealed. The DFT for a speech signal, $x[n]$, is defined as:

$$x[k] = \Sigma_{n=0}^{N-1} x[n] e^{-j\frac{2\pi kn}{N}}, k = 0, 1..., N-1 \tag{2.4}$$

The time complexity of DFT computation is N^2 which makes it difficult to use. There is a family of algorithms called Fast Fourier Transform (FFT) with $Nlog_2N$ time complexity that is a faster implementation of DFT [31].

5. **Mel Filter Bank:** When the speech frequency increases, the human perceived frequency decreases to make the human voice perception less sensitive to frequency. The Mel Filter works in the same manner. Mel filter bank is a band-pass filter that uses more narrow-bandwidth filters at low frequencies and less wide-bandwidth filters at high frequencies. The magnitude spectrum $x[k]$ will be transformed to Mel scale with:

$$S[l] = \Sigma_{k=0}^{\frac{N}{2}-1} H_l[k] x[k], l = 1, 2, ..., L \tag{2.5}$$

where $H_l[k]$, is the lth Mel-filter.

6. **log:** A logarithm is taken to suppress the high amplitudes. The output of this step is known as *filter banks* and they are calculated as:

$$S^l[l] = log(S[l]), l = 1, 2, ..., L \tag{2.6}$$

where $S^l[l]$ are called Mel scale filter banks.

7. **DCT:** Discrete Cosine Transform (DCT) is applied to decorrelate the filter bank representation:

$$y[p] = \frac{1}{2}\Sigma_{l=1}^{L} S^l[l]\cos(\frac{\pi p(l-0.5)}{L}), p = 0, 1, ...j-1 < L \tag{2.7}$$

8. **MFCC:** The resulting cepstral coefficients from the previous step are called Mel Frequency Cepstral Coefficients (MFCC) features. Normally the first coefficients are used and the remaining will be discarded.

9. Δ **and** $\Delta\Delta$**:** The first and second derivations of MFCCs

The log Mel scale filter bank and MFCCs are used in the speaker recognition systems. The steps of extracting the filter bank and MFCCs are shown in Figure 2.1. The Mel scale filter bank is obtained by Equation 2.6. The MFCCs are the output of DCT transformation and they are obtained by Equation 2.7. In speaker recognition systems it is common to use the first 19 coefficients of DCT with the log energy of the frame. These 20 features are concatenated with their first and second derivations, Δ, $\Delta\Delta$, produces the 60 coefficient features for each frame [32].

$$MFCC = \{c_1, ...c_{19}, e, \Delta c_1, ..., \Delta c_{19}, \Delta e, \Delta\Delta c_1, ...\Delta\Delta c_{19}, \Delta\Delta e\} \tag{2.8}$$

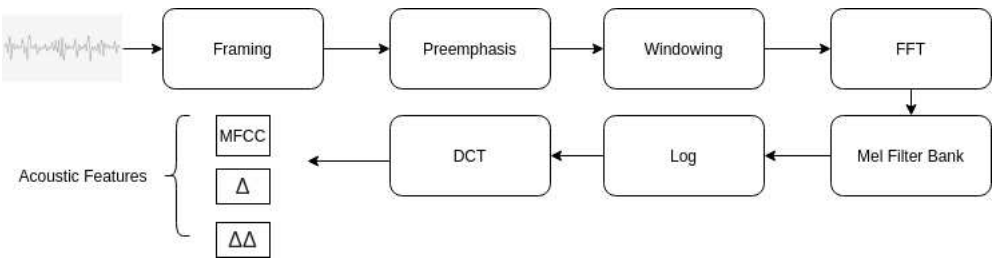


Figure 2.1: The steps of Mel filter bank and MFCC feature extraction

Since the acoustical variabilities have not been targeted during filter bank and MFCC feature extraction, the speaker recognition systems trained with them are not robust and their performance reduces in the presence of variabilities. Several approaches such as mapping-based and adversarial training methods are used to reduce the impact of variabilities on filter banks and MFCC features. These approaches will be discussed in Section 3.3. Before moving to the DNN speaker embedding systems, we will discuss unsupervised neural speech representations that are more robust against noise and reverberation.

2.2.3. UNSUPERVISED NEURAL SPEECH REPRESENTATION

The unsupervised methods based on deep learning are replacing hand-crafted features such as MFCCs. The goal of unsupervised learning methods is to learn a speech representation from unlabeled raw speech that can be used in a downstream task

such as speaker recognition, automatic speech recognition, etc. There are several unsupervised speech representation models such as Wav2Vec and WavLM.

The Wav2Vec model is composed of encoder and contextualization networks. In this model, the encoder maps the raw speech signal, χ, into a latent representation, $z_1, ..., z_T$ with a stack of convolutional layers. The output of the encoder is given to the contextualization network, $g : \zeta \longmapsto \theta$, in order to capture the context information for the latent space representation. The network is trained by optimizing a contrastive binary classification task. The Wav2Vec predicts for z_i the correct representation of z_{i+k} next frames. The final loss is the summation of the contrastive loss calculated for predicting $1, 2, ..., K$ next frames. Instead of using MFCC or Filter banks, the output of the contextualization network can be used as speech representation features in downward tasks such as speaker recognition systems. The architecture of Wav2Vec is shown in Fig. 2.2.

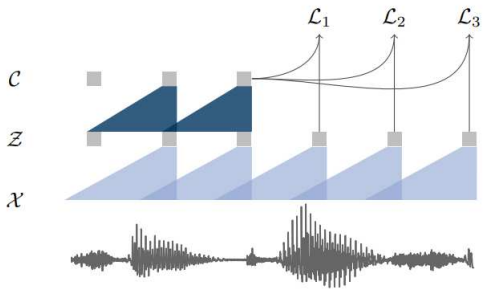


Figure 2.2: The architecture of Wav2Vec speech representation systems [33]

Wav2Vec2 is a modified version of Wav2Vec. Besides encoder and contextualization networks, the Wav2Vec2 has a quantization module $q : \zeta \longmapsto \omega$ that discretizes speech representation in latent space to a finite set of representations. The encoder is composed of stacked convolutional layers and the contextualization network uses the Transformer architecture. The network is trained to predict the correct quantized speech representations. The architecture is shown in Figure 2.3. It is shown that Wav2Vec2.0 gives competitive results in comparison to known benchmarks in speaker recognition [34].

The WavLM speech representation model is composed of a CNN encoder and a Transformer architecture. First, a partially noisy raw speech is given to the encoder. For a given speech signal two types of noises are used for data augmentation. The first type is noises chosen randomly from a noise pool. The second type of noise is a random speech file chosen from the same mini-batch. In both cases, the noise

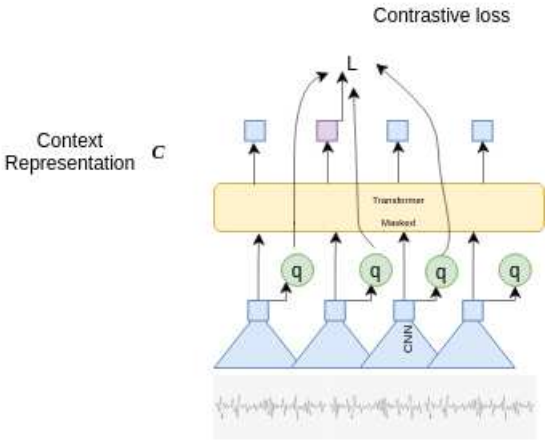


Figure 2.3: The architecture of Wav2Vec2 speech representation systems [35]

is added to the utmost 50% of the clean speech signal. Since the speech noise comes from the same speaker in the same mini-batch, it can make the task of prediction more difficult which results in more speaker discriminant representation. Second, the Transformer is fed with the masked version of the encoded speech. The WavLM predicts the masked parts of the input. In the masked prediction task, the model is trained to identify the main speaker from the noisy/overlapped speech and predict the content information corresponding to the main speaker. The architecture of WavLM is shown in Figure 2.4. The results on the speaker recognition task with WavLM outperform the hand-crafted features significantly [36]. In this pipeline, the weighted sum of all transformer layers is used as input features for training the speaker embedding network.

2.3. FROM STATISTICAL TOWARDS DNN-BASED SPEAKER EMBEDDING

Speaker modeling is the core component of a speaker recognition system. The goal of a speaker modeling system is to find a compact fixed-length speaker discriminant representation for a variable-length utterance. In this subsection, we will present a short history of statistical speaker embedding systems and a transition towards DNN-based speaker embedding systems.

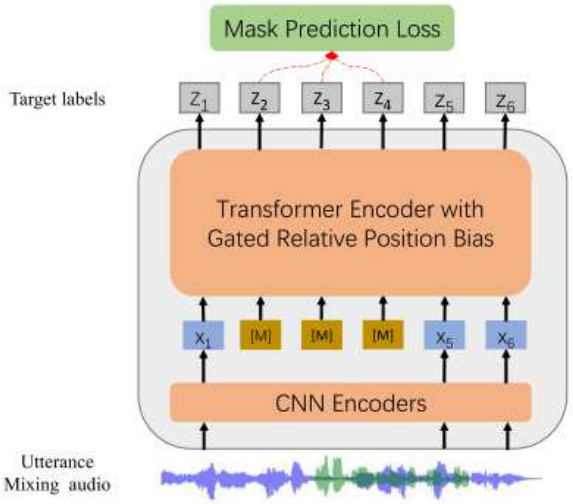


Fig. 1. Model architecture.

Figure 2.4: The architecture of WavLM speech representation framework [36]

2.3.1. STATISTICAL SPEAKER MODELING SYSTEMS

The i-vector framework is the latest and the most successful GMM-based speaker modeling system. It has evolved from two precedent systems: joint factor analysis (JFA) [37] and GMM-SVM [38] systems. These models are based on the concept of super-vectors. The super-vectors are the concatenation of Gaussian mean parameters estimated by the maximum apriori adaptation. In JFA, it is assumed that for a given utterance from a specific speaker, $m_{s,h}$, the GMM super-vector is the linear combination of four elements [39]:

$$m_{s,h} = m_0 + Vy + Ux + Dz \tag{2.9}$$

where m_0 denotes speaker/channel/environment-independent information, V and D are speaker subspace, U is session/environment-dependant subspace. The y,z, and x are the speaker and session-dependent factors in their respective subspace. Since it was observed that the channel component contains speaker information, a new formulation was proposed that combines these components in a single space called total variability. In this new formulation the GMM super-vector for $m_{s,h}$ is:

$$m_{s,h} = m_0 + Tw_{s,h} \tag{2.10}$$

where $w_{s,h} \sim \mathcal{N}(0, I)$ is called total factors. The total factors are hidden variables estimated by their posterior expectation and are called *i-vectors*. In the scoring step, *i-vectors* are used as fixed-length representations for each speech utterance [16].

2.3.2. BOTTLENECK FEATURES

Bottleneck features can be considered as a bridge between statistical and DNN-based speaker modeling systems. The bottleneck features are transformed versions of spectral features by a neural network. In [40] a feed-forward network is proposed that accepts the MFCCs at the input and classifies speakers. A hidden layer representation is used as a bottleneck feature for training the GMM-UBM speaker recognition system. Since the neural network is trained with utterance-level MFCCs, the bottleneck features can capture the information at both the local and global levels of a speech utterance. The bottleneck features sparked a line of research and different implementations with statistical speaker modeling systems including with i-vector systems are explored [41].

2.3.3. D-VECTORS

A big leap in the evolution of the DNN-based speaker recognition systems is the d-vector that brought us closer to the general framework of the current DNN-based speaker recognition systems [1]. The d-vector system is a fully-connected feed-forward neural network that accepts the filter bank features with a context for each frame and performs a speaker classification task. The motivation behind this idea is this assumption: the latent representation obtained by a speaker classifier DNN captures the speaker's characteristics. The authenticity of this assumption is the key to the success of DNN-based speaker recognition systems. The DNN-based speaker recognition systems are more complicated versions of the d-vector system. A trained d-vector network with development data can be used to extract speaker embeddings for enrolment and test utterances. During the scoring phase, the output layer is removed and the average of the embedding layer for all input frames is used as a compact speaker representation for a speech signal. The architecture of d-vector systems is shown in Figure 2.5.

2.4. DNN-BASED ARCHITECTURES

In this section, the main DNN-based speaker embedding networks are outlined. The general architecture of DNNs used for speaker recognition systems is comprised of frame-level feature representation, pooling layer, speaker embeddings, and speaker classifier:

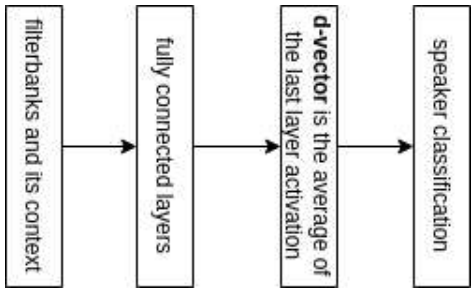


Figure 2.5: The d-vector speaker embedding extractor

- Input features can be raw signals in the time domain, acoustic features such as the spectrogram, filter bank, or MFCC, or DNN-based unsupervised speech representations such as Wav2Vec or WavLM.
- Frame level layers transform the input features to latent space.
- Temporal pooling performs a transformation from frame-level embedding features to utterance-level embedding features.
- Speaker embedding layer is a fully-connected layer that gives the aggregated representation obtained from the pooling layer.
- Speaker classifier performs a multi-class speaker classification task by optimizing an objective function such as softmax or its variants.

The general architecture of the DNN speaker embedding network is depicted in Figure 2.6.

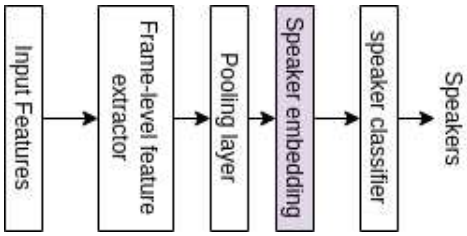


Figure 2.6: The components of a DNN speaker embedding network

In the following subsections, all modules of DNN-based speaker embeddings are discussed.

2.4.1. TIME DELAY NEURAL NETWORKS (TDNN)

Time Delay Neural Networks (TDNN) are feed-forward neural networks that model the context of speech frames efficiently. In a normal feed-forwarded neural network each computational unit accepts the weighted sum of the input speech frame, while in the TDNN the computational units accept the weighted sum of the speech frames alongside the weighted sum of the context (Time Delay). For example, in a normal feed-forward neural network the first hidden layer neurons accept the weighted sum of $i_1, ...i_n$ features, but in the TDNN with delay $D = 2$ it accepts the weighted sum for $D = 1$ and $D = 2$. It means that the number of inputs for the first hidden layer neurons increases $D + 1$ times [42].

This characteristic of TDNN allows the deeper layers to receive speech signals across multiple frames. For example, in Figure 2.7 for a specific frame, t, the first hidden layer receives five frames $[t-2, t+2]$; the second hidden layer receives three frames at $t-2$, t, and $t+2$ from the first hidden layer; the third hidden layer receives three frames at $t-3$, t, and $t+3$ from the second hidden layer. At the third hidden layer, each neuron captures 15 frames [43].

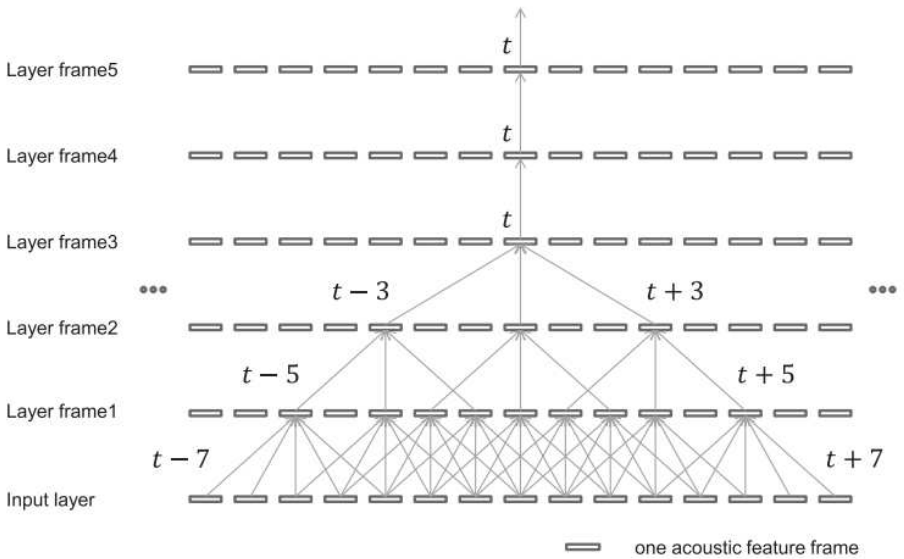


Figure 2.7: TDNN context capturing[43]

The architecture of TDNN-based speaker embedding is composed of several TDNN

hidden layers that extract frame-level features. After TDNN layers there is a statistics pooling layer that aggregates information through different frames in order to create a compact fixed-length representation. The next layer is speaker embedding which is called the x-vector [2]. After the speaker embedding layer there is a fully connected speaker classification network. The architecture of the TDNN speaker embedding network is shown in Figure 2.8.

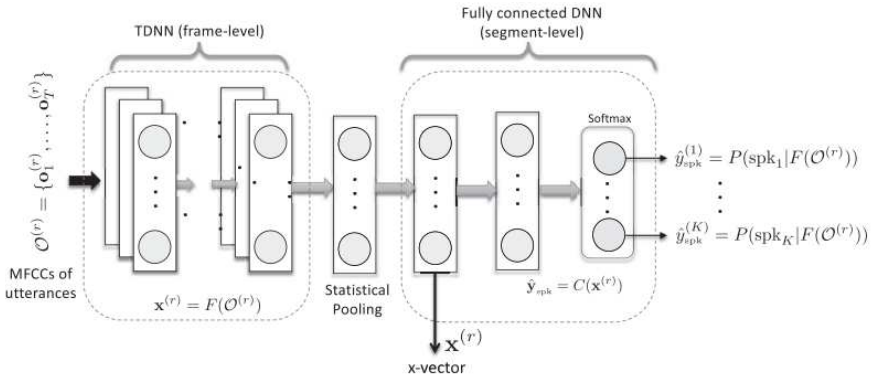


Figure 2.8: The architecture of TDNN speaker embedding extractor network [43]

2.4.2. RESNET

Convolutional neural networks (CNN) are used widely in speaker embedding extractors. The CNNs are leveraging sparse interaction, and weight sharing to reduce the computational cost and increase efficiency. Also, they can be used for working with variable-length inputs such as speech signals. In a convolution layer, three operations are accomplished: a set of parallel convolution operations to generate a linear activation, passing the linear activation through a nonlinear function such as ReLu, and the pooling operation that produces a statistic summary from the network output [44]. The ResNet speaker embedding network is a stack of convolutional layers that are based on VGG networks for image classification [45].

With more than 30 layers, ResNet is a very deep network. A main obstacle on the way of training a very deep neural network is vanishing/exploding gradients, which prevent the convergences of training algorithms. Adding more layers necessarily doesn't increase the performance of DNNs and after some iterations cause performance degradation. Residual neural networks are a successful solution to this problem [46].

The hypothesis behind this network is that estimating a residual mapping is easier than estimating the original one. For $H(X)$ a function that should be estimated by a neural network, residual networks estimate:

$$F(x) = H(x) - x \tag{2.11}$$

then

$$H(x) = F(x) + x \tag{2.12}$$

This task is done by a shortcut connection in residual blocks shown in Figure 2.9:

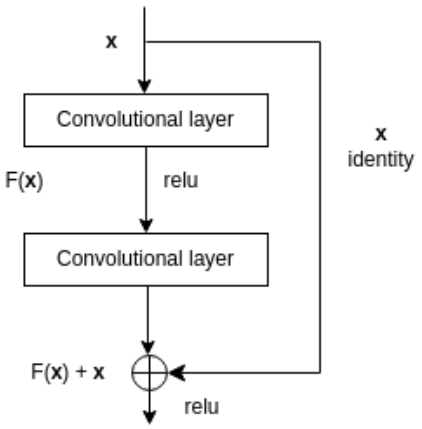


Figure 2.9: ResNet block [46]

The ResNet speaker embedding system is a convolutional neural network composed of stacking ResNet blocks. The shortcut connections are identity functions. If the size of $F(x)$ and x is not the same, the size of shortcut connections will be increased by zero padding.

According to the number of layers, there are several common architectures for ResNet systems including ResNet34, ResNet50, ReNet101, etc. In this thesis, we are using ResNet34 which is composed of 34 layers. The layers are separated according to the size of the feature map and the number of filters. In all convolution layers, a 3×3 convolution filter is used. In deeper layers where the size of feature maps is divided by 2, the number of filters is multiplied by 2. All convolutional layers are followed by ReLu and batch normalization layers. After the convolution layer, a statistics pooling layer is used and a softmax fully connected layer attributes the speaker class to the input signal. The architecture of ResNet34 is shown in Figure 2.10

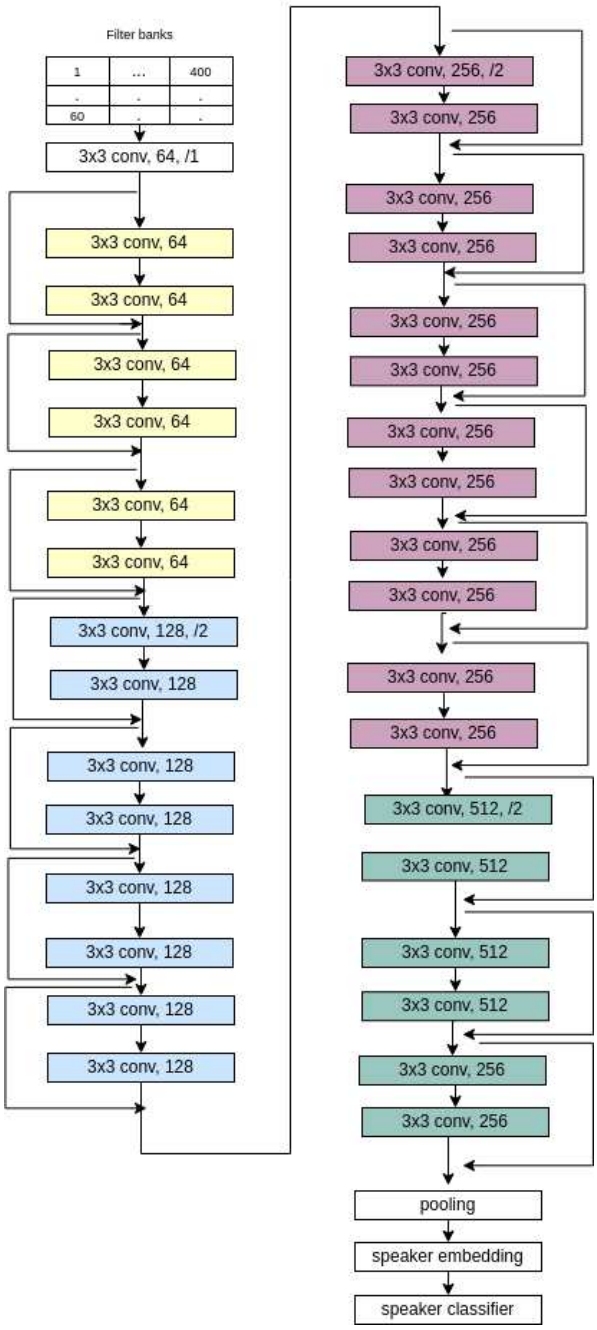


Figure 2.10: The architecture of ResNet speaker embedding extractor

2.4.3. ECAPA TDNN

The ECAPA-TDNN is an altered version of the ResNet system. This system is composed of ECAPA blocks. The ECAPA blocks rescale frame-level features with squeeze and excitation blocks. Each ECAPA block is composed of a convolutional layer sandwiched between a dense preceding convolution layer for dimension reduction and another dense layer to regenerate the input dimension. The output of the third layer is given to a squeeze and excitation block that gives different weights to each channel. Similar to the ResNet architecture there is a shortcut connection from the input of each block to the output (Figure. 2.11).

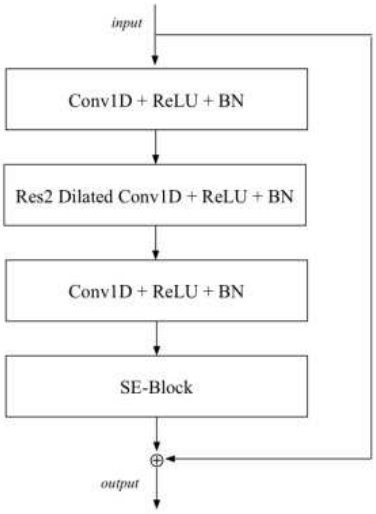


Figure 2.11: ECAPA block composed of a dilated convolutional layer located between two dense layers followed by e Squeeze-and-Excitation block to rescale frames and a shortcut connection from the input to the output of the block [5]

Despite ResNet, in ECAPA-TDNN first dimensional convolutional layers are used [5]. Since the empirical experiments in the domain of DNN-based speaker embeddings show that both shallow and deep features can contribute to achieving better speaker representations, hierarchical feature learning is done by aggregation and propagation of features at different hierarchical levels. The concatenated multi-level features (MFA) are given to attentive statistics pooling, another contribution of the ECAPA-TDNN pipeline, that produces utterance-level representation. This pooling strategy is described in Section 2.4.5.

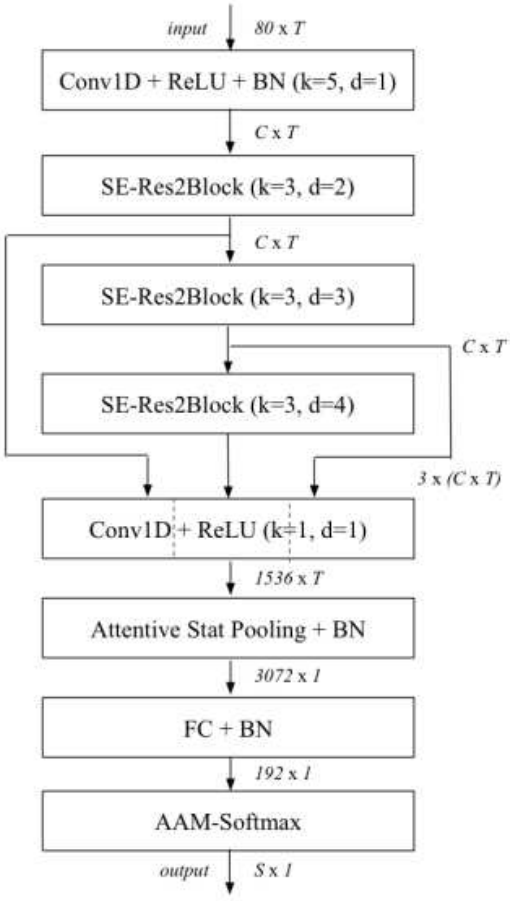


Figure 2.12: ECAPA-TDNN speaker embedding network composed of ECAPA blocks and multi-level hierarchical feature propagation connection, k and d stand for kernel size and d for dilation spacing parameters of convolutional dilation layers, C, T, and S correspond to channels, features dimension and number of speakers, FC is a fully connected layer followed by BN batch normalization[5]

2.4.4. MFA-CONFORMER

Multi-scale Feature Aggregation Conformer (MFA-Conformer) is a speaker embedding architecture that leverages Convolution-augmented Transformer (Conformer) at the frame level. The basic block of this system is the Conformer block [47]. The idea behind using the Conformer block is to capture both local features and global features.

To achieve this goal, the Conformer block uses the capability of convolutional layers and Transformers. The convolutional layers are strong in capturing local features but they can not capture the global dependencies which is the inverse in the case of Transformers. The Transformers can capture global features successfully but they are not strong in capturing local features. The Conformer block is composed of two feed-forward neural networks a multi-head self-attention layer and a convolution layer. The MFA-Conformers hire the advantages of both convolution layers and Transformers [4]. For h_i a given frame the output of the conformer block h_o is calculated as Equation 2.13.

$$\begin{aligned}
\hat{\mathbf{h_i}} &= \mathbf{h_i} + \frac{1}{2}FNN(\mathbf{h_i}) \\
\mathbf{h_i}' &= \hat{\mathbf{h_i}} + MHSA(\hat{\mathbf{h_i}}) \\
\mathbf{h_i}'' &= \mathbf{h_i}' + Conv(\mathbf{h_i}') \\
\mathbf{h_o} &= \mathbf{h_i}'' + \frac{1}{2}FNN(\mathbf{h_i}'')
\end{aligned} \tag{2.13}$$

where FNN stands for feed-forward neural network, and $MHSA$ is multi-headed self-attention network, $\hat{\mathbf{h_i}}$ is the input of MFA-conformer block and $\mathbf{h_o}$ is the output of MFA-conformer block.

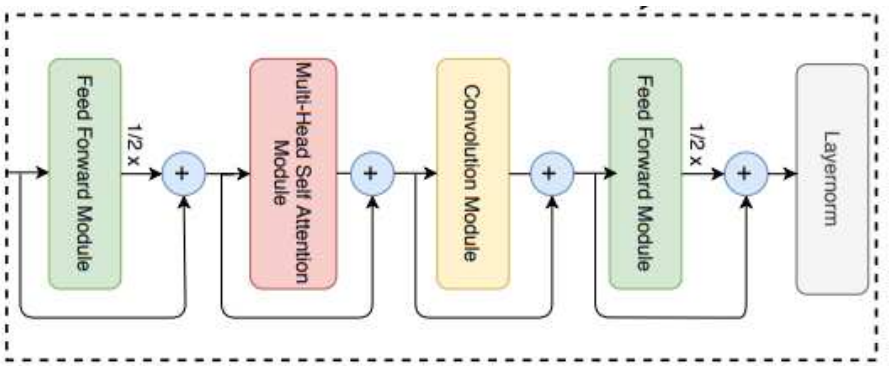


Figure 2.13: MFA-Conformer block [47]

The input features of the MFA-Conformer are given to a subsampling convolution layer in order to reduce the computational load. The output will be given to N Conformer blocks. The output of all Conformers blocks will be concatenated and after normalization, it will be given to the attentive pooling layer (Figure [4]).

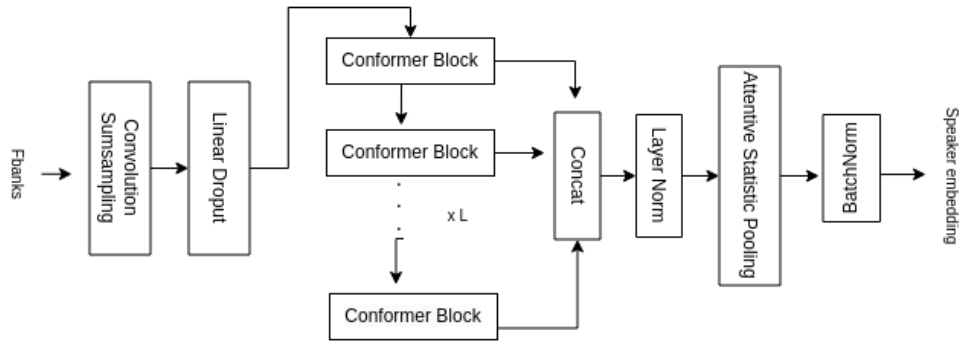


Figure 2.14: MFA-Conformer architecture [4]

2.4.5. POOLING LAYER

The temporal pooling operation converts the frame-level features into utterance-level representation. For a speech signal $H = \{h_t \in R^{d_2} | t = 1, 2, ..., T\}$, where h_t is the tth features given by frame-level feature extractor, the utterance-level representation, $\mathbf{u}$, can be defined in several ways as follows:

AVERAGE POOLING

Average pooling is the simplest and most widely used pooling function. It is defined as the average of frame-level features:

$$\mathbf{u} = \frac{1}{T}\Sigma_1^T h_t \tag{2.14}$$

STATISTICS POOLING

The statistics pooling is the concatenation of the statistic mean, m, and, the standard deviation , σ, of H. The statistic mean is defined as:

$$\mu = \frac{1}{T}\Sigma_1^T h_t \tag{2.15}$$

and the standard deviation is

$$\sigma = \sqrt{\Sigma_{t=1}^T \frac{1}{T}(h_t \odot h_t - \mu \odot \mu)} \tag{2.16}$$

where $\odot$ is the element-wise product. From Equation 2.15 and 2.16 the statistics pooling is defined as:

$$\mathbf{u} = [\mu^T . \sigma^T]^T \tag{2.17}$$

ATTENTIVE POOLING

Both average and statistics pooling gives the same weight to all frames in a speech utterance. However, some frames maybe contribute more to having better representations. Attention strategy gives bigger weight to frames that contribute more to having a more speaker-discriminant representation [32]. For H frames belonging to a specific utterance, the attention can be regarded as a weight vector corresponding to each frame. The attentive statistics pooling is a weighted version of statistics pooling [4]. For h_t, a given frame at time t, a scale score can be calculated as

$$e_t = v^T f(\mathbf{W}h_t + b) + k \tag{2.18}$$

where $W \in \mathbb{R}^{R\times D}$, $b \in \mathbb{R}^{R\times 1}$, $\mathbf{v} \in \mathbb{R}^{R\times 1}$, and k are trainable parameters. For calculating α_i, the scale score, a normalization score can be calculated as:

$$\alpha_t = \frac{exp(e_t)}{\Sigma_\tau^T exp(e_\tau)} \tag{2.19}$$

The weighted mean is defined as:

$$\hat{\mu} = \Sigma_1^T \alpha_t h_t \tag{2.20}$$

and the weighted version of the standard deviation is defined as:

$$\hat{\sigma} = \sqrt{\Sigma_{t=1}^T \alpha_t (h_t \odot h_t - \mu \odot \mu)} \tag{2.21}$$

where $\mu = \frac{1}{T}\Sigma_1^T h_t$ and $\odot$ is point-wise product. The attentive pooing is the concatenation of $\hat{\mu}$ weighted mean, and $\hat{\sigma}$ weighted standard deviation.

2.4.6. OBJECTIVE FUNCTIONS

The last part of DNN-based speaker recognition systems is the speaker classifier. The speaker classifier is a feed-forward multiclass classification network. At the output layer of the speaker classification network, the loss value is calculated based on an objective function. Both generator and speaker classifier networks are trained in order to reduce this loss value. The objective function can impact the performance of speaker embedding in general and in specific cases such as the presence of variability. In this subsection, the common objective functions used in speaker recognition systems are presented.

SOFTMAX

For a multi-class classification task, the cross-entropy is calculated as

$$CE = -\frac{1}{N}\Sigma_{n=1}^{N}\Sigma_{j=1}^{J} t_{nj} log\ p_{nj} \tag{2.22}$$

where $(t_{nj} = 1) \iff x_n \in class\ c_j$, p_{nj} is the posterior probability for x_n belonging to c_j, and N is the number of training data points. If softmax is used as the activation function:

$$p_{nj} = \frac{exp(\mathbf{w}_j^T \mathbf{x}_n + b_j)}{\Sigma_{j=1}^{J} exp(\mathbf{w}_j^T \mathbf{x}_n + b_j)} \tag{2.23}$$

By replacing 2.23 in 2.22 we achieve to softmax loss function

$$Softmax = -\frac{1}{N}\Sigma_{n=1}^{N} log\ \frac{exp(\mathbf{w}_{l_n}^T \mathbf{x}_n + b_j)}{\Sigma_{j=1}^{J} exp(\mathbf{w}_j^T \mathbf{x}_n + b_j)} \tag{2.24}$$

The Softmax loss function is used with d-vector and x-vector pipelines discussed in section 5.2.1.

ANGULAR SOFTMAX (ASOFTMAX) LOSS

The angular softmax is a modified version of softmax that separates samples in the angular space and adds an angular margin between classes [48]. If we rewrite the dot product of $\mathbf{w_j^T x_n}$ in the following form

$$\mathbf{w_j^T x_n} = \|\mathbf{w_j}\| \cdot \|\mathbf{x_n}\| \cos(\theta_{j,n}) \qquad 0 \leq \cos(\theta_{j_n,n}) \leq \pi \tag{2.25}$$

After normalizing the weights, putting bias to zero, and adding a margin to the angle with $\phi(\theta_{j,n}) = (-1^k)\cos(m\theta_{j,n}) - 2k$ a monotonic function where $m \geq 1$, $\theta_{j,n} \in [\frac{k\pi}{m}, \frac{(k+1)\pi}{m}]$, and $k \in \{1, 2..., m\}$, the Equation 2.24 can be rewritten as:

$$L_{AS} = -\frac{1}{N}\Sigma_{n=1}^{N} \log \frac{exp(\|\mathbf{x_n}\|\phi(\theta_{j,n}))}{Z} \tag{2.26}$$

where

$$Z = exp(\|\mathbf{x_n}\|\phi(\theta_{j,n})) + \Sigma_{j \neg \theta_{j,n}} exp(\|\mathbf{x_n}\| \cos(\theta_{j,n})) \tag{2.27}$$

where m is an integer value and is called a margin hyperparameter.

A more flexible version of angular softmax are Additive margin softmax (AMSoftmax) loss [49] and Large margin softmax loss [50] for speaker verification. The ASoftmax, AMSoftmax, and LMSoftmax improve the speaker embeddings in two ways. Firstly the

embeddings are angularly redistributed. This characteristic makes them more suitable with the cosine distance back-end discussed in Section 2.5.3. Furthermore, they consider both inter-class and intra-class distances to be optimized while the cross entropy just increases the inter-class distance [32].

2.5. BACKENDS

The last part of a speaker recognition system is scoring. It is common to reduce the number of speakers embedding dimensions and project them to a more separable space by a dimensionality reduction technique. In this section, the LDA projection is described. After that, two main scoring techniques (i.e. PLDA and cosine similarity) are described.

2.5.1. LDA

Linear discriminant analysis (LDA) is a dimensionality reduction technique that is applied before scoring. The LDA works based on an optimization problem that projects the speaker embeddings coming from different speakers into a lower dimension space by maximizing the ratio of inter-class scatter over intra-class scatter [51]. If we consider the S_W as intra-class covariance and S_B as inter-class covariance matrix the LDA tries to optimize the following equation with an optimization method such as maximum likelihood [52]:

$$\underset{W}{\operatorname{argmax}} J(W) = \frac{W^T S_B W}{W^T S_W W} \tag{2.28}$$

Before scoring with PLDA, it is common to reduce the dimensionality and increase the discriminability between speaker embeddings belonging to different speakers.

2.5.2. PLDA

Probabilistic Linear Discriminant Analysis (PLDA) is one of the most popular back-ends used for scoring in speaker recognition systems. The PLDA accepts pairs of speaker embeddings and does the scoring based on log-likelihood ratio (LR) hypothesis testing. For x_s and x_t given speaker embeddings, assume:

H_0: x_s and x_t comes from the same speaker

H_1: x_s and x_t comes from distinct speakers

The PLDA produces the likelihood ratio score for x_s and x_t given speaker embeddings:

$$S_{LR}(x_s, x_t) = \frac{p(x_s, x_t|H_0)}{p(x_s, x_t|H_1)} \tag{2.29}$$

If the $S_{LR}(x_s, x_t)$ will be greater than the threshold, then they come from the same speaker, otherwise, the speakers are distinct. The PLDA assumes that the general format of speaker embeddings is:

$$x = m + Vz + \epsilon \tag{2.30}$$

where m is an global mean of all speaker embeddings, V is the speakers loading matrix, z is the speaker specific factor, and ϵ is the residual information including channel information.

From Equation 2.29, and Equation 2.30 we will get:

$$\begin{aligned} log S_{LR}(x_s, x_t) = log\mathcal{N}(\begin{bmatrix} x_s \\ x_t \end{bmatrix} \begin{bmatrix} m \\ m \end{bmatrix} \begin{bmatrix} \Sigma_{tot} & \Sigma_{ac} \\ \Sigma_{ac} & \Sigma_{tot} \end{bmatrix}) \\ -log\mathcal{N}(\begin{bmatrix} x_s \\ x_t \end{bmatrix} \begin{bmatrix} m \\ m \end{bmatrix} \begin{bmatrix} \Sigma_{tot} & 0 \\ 0 & \Sigma_{tot} \end{bmatrix}) \end{aligned} \tag{2.31}$$

where $\Sigma_{tot} = VV^T + \Sigma$ and $\Sigma_{ac} = VV^T$. Because m is the average of all speaker embeddings, it can be computed once. After removing it from all speaker embeddings in Equation 2.31 it will be replaced by 0, by doing that and expanding Equation 2.31 we will have:

$$log S_{LR}(x_s, x_t) = \frac{1}{2}[x_s^T Q x_s + 2x_s^T P x_t + x_s^T Q x_t] + const \tag{2.32}$$

where

$$\mathbf{Q} = \Sigma_{tot}^{-1} - (\Sigma_{tot} - \Sigma_{ac}\Sigma_{tot}^{-1}\Sigma_{ac})^{-1} \tag{2.33}$$

and

$$\mathbf{P} = \Sigma_{tot}^{-1}\Sigma_{ac}(\Sigma_{tot} - \Sigma_{ac}\Sigma_{tot}^{-1}\Sigma_{ac})^{-1} \tag{2.34}$$

The presented formulation is for the case of having one enrollment per speaker [43]. In the case of having more than one enrollment per speaker, the speaker embeddings averaging (take the average of all speaker embeddings per speaker) or score averaging (take the average of scores for all enrollments per speaker) can be used. In this thesis, we are using score averaging.

2.5.3. COSINE SIMILARITY SCORING

The cosine similarity scoring is a simple and efficient scoring technique in the speaker embedding space. For a pair of x_s and x_t speaker embeddings, the cosine similarity is:

$$S_{cs}(x_s, x_t) = \frac{x_s.x_t}{\|x_s\| \times \|x_t\|} \tag{2.35}$$

The cosine similarity ranges between -1 for completely opposite speaker embeddings and 1 for exactly the same speaker embeddings.

The motivation behind using PLDA scoring is channel compensation but in DDN-based speaker embeddings, the channel variability can be compensated relatively in the speaker embeddings space by training a robust speaker embeddings extractor or data augmentation [53, 54]. This characteristic helps us to replace the PLDA with cosine similarity. The second reason that makes the cosine similarity more plausible is the redistribution of speaker embeddings by the angular softmax objective function or its variants.

2.6. EVALUATION METRICS

Before defining the evaluation metrics, we will discuss the types of errors in speaker recognition systems. Each speaker recognition system faces two types of attempts (trials) named target (real-speaker) and non-target (imposters). In the target trials, the claimed and the registered users are the same but in the non-target trials, the registered speaker and the claimed user are different. According to these definitions, there are two types of errors:

- False acceptance(FA) or (False alarm): grant access to imposter speaker
- False rejection (FR) or (Miss error): denying access to a legitimate speaker

From the definition of FA and FR, the error rates are defined as

$$False\ acceptance\ rate = \frac{Number\ of\ FA\ errors}{Total\ number\ of\ imposters} \tag{2.36}$$

and,

$$False\ rejection\ rate = \frac{Number\ of\ FR\ errors}{Total\ number\ of\ legitimate\ attempts} \tag{2.37}$$

If we consider the speaker recognition task as a binary classification, types of errors can be rewritten in terms of a confusion matrix, as Table 2.1:

Table 2.1: Types of errors in SR systems

	Accept	Reject
Accept	True Accept	False Accept
Reject	False Reject	True Reject

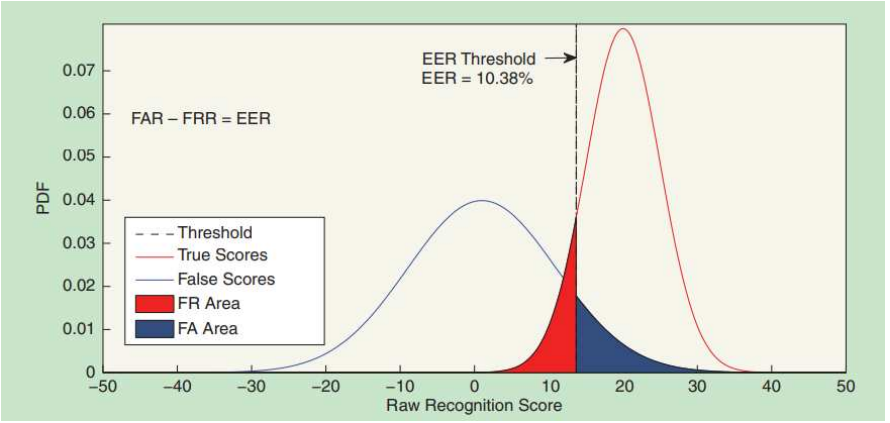


Figure 2.15: The EER evaluation metric [16]

2.6.1. EQUAL ERROR RATE

The Equal Error Rate (EER) is defined as an operating point θ, where $FAR = FRR$; moving it from left to right reduces the FAR errors from 100 toward 0 and increases the FRR from zero to 100. This process is shown in Figure 2.15. Choosing a smaller θ means that the system accepts more illegitimate users and it is less secure, meanwhile, a high threshold reduces the FAR which means the system is robust but it is not user-friendly. The EER is a point in which there is a tradeoff between user-friendliness and robustness. A normalized version of EER is called HRR which is defined as $EER/2$.

2.6.2. DET

When we want to compromise between two types of FAR and FRR errors in different operating points, the EER is not sufficient. In this case, the Detection Error Trade-off (DET) curve is a common metric [55]. The DET curve is the plot of both FAR and FRR across two different axes which is shown in Figure 2.16.

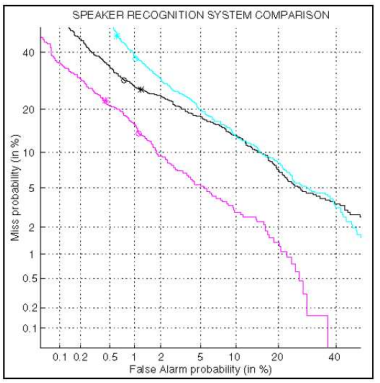


Figure 2.16: Detection error tradeoff curve

2.6.3. DETECTION COST FUNCTION

The Detection Cost Function (DCF) is a weighted sum of FRR ($P_{Miss|Target}$) and FAR ($P_{FalseAlarm|NonTarget}$) [56]. The DCF can be calculated as:

$$DCF(\theta) = C_{Miss} \times P_{Miss|Target}(\theta) \times P_{Reject} + C_{False\ Alarm} \times P_{False\ Alarm|NonTarget}(\theta) \times (1 - P_{Target}) \tag{2.38}$$

where: θ is a decision threshold, C_{Miss} is the cost of false rejection, $C_{False\ Alarm}$ is the cost of false acceptance, P_{Target} is the prior probability of target speakers.

2.7. CONCLUSION

In this chapter, we reviewed the current DNN-based speaker recognition systems. Firstly we describe the feature extraction procedure. We described the MFCC and filter banks. We mentioned that during the extraction of these features, the impact of noise and reverberation is not considered. In order to reduce the impact of these variabilities a speech enhancement method can be applied while the state-of-the-art speech representation methods such as Wav2Vec or WavLM can target the impact of environmental variabilities such as additive noise.

In the second part of this chapter, we showed the evolving path of the DNN-based speaker embedding extractors. The x-vector (TDNN) system captures the speech context in comparison to its predecessor, the d-vector, and the pooling operation over frame-level representation gives more robust speaker embeddings. The ResNet system uses the idea of residual connection to help the training of very deep networks.

In the ECAPA-TDNN system, the residual architecture is equipped with squeeze and excitation. The MFA-Conformer tries to exploit the speakers' representation at different levels. Also, it captures both local and global speaker features by means of a combination of convolutional layers and transformer.

The PLDA and cosine similarity metrics are used as scoring techniques in this thesis. Finally, the common evaluation metrics including the EER, DCF, and DET curve described. In the next chapter, the different levels of handling noise and reverberation in speaker embedding systems will be described.

3

SPEAKER RECOGNITION CHALLENGES AND ADAPTATION

This chapter is devoted to speaker recognition variabilities and different methods used to suppress their negative impact. Data augmentation is the main approach for training noise robust speaker embeddings. We first describe the data augmentation approaches. It is possible to make the speech recognition systems robust against variabilities in different levels of the system. We will start with speech enhancement methods. Among the speech enhancement methods, we review the methods based on DNNs including masking-based, mapping-based, and adversarial training approaches. Different strategies for making speaker embedding extractors robust against variabilities such as adversarial training, discrepancy methods, pooling strategies, and attention mechanisms are reviewed. The speaker embedding transformation comes after.

3.1. INTRODUCTION

THE general framework of DNN-based speaker embedding doesn't target explicitly the negative impact of existing variabilities and domain mismatches. Robust speaker recognition refers to different approaches that target explicitly the negative impact of different types of variabilities. This chapter is devoted to robust speaker recognition.

In Chapter 1 different types of variabilities including internal, external, and conversation-based variabilities described, and Chapter 2 reviewed DNN-based speaker recognition architectures. In this chapter, we will review the main approaches to make DNN-based speaker recognition systems more robust against variabilities. The robust speaker recognition approaches can be categorized from different points of view. We will categorize them in terms of the level at which the compensation is done. In speaker recognition systems domain adaptation can be done at the following levels: speech signal, feature extraction, speaker modeling, and speaker embedding and/or data augmentation techniques. We will concentrate on the methodology and for each method, the applications for specific variabilities will be given.

The first group of robust speaker recognition approaches is data augmentation. Data augmentation refers to increasing the size and diversity of training data by means of constructing new images for a speech signal. Adding noise and reverberation at the signal level or applying time-frequency masks at the feature level are among common data augmentation techniques. In Section 3.2 the application of data augmentation in robust speaker recognition is described.

The second group is speech enhancement methods at signal level or feature level which are discussed in Section 3.3. The speech enhancement methods are working at the front end of the speaker recognition system by removing the impact of acoustical variabilities on raw speech signals or spectral features. In this chapter three groups of speech enhancement methods are reviewed: masking-based, mapping-based, and adversarial training. The masking-based speech enhancement methods try to reconstruct the clean version of the speech signal in the time-frequency domain. The mapping-based and adversarial-based methods are deeply investigated at the feature level.

The third group is robust speaker modeling approaches; making the speaker embedding network robust to interferences by proposing robust architecture, robust objectives, or robust learning methods. Several strategies such as adversarial training, multitask training, and discrepancy-based regularizers (e.g. Coral, MMD, KL) can be used to produce robust speaker embeddings. Exploring more robust pooling operation and using attention mechanism are among other modifications of DNN speaker

embeddings reviewed in Section 3.4

The speaker embedding transformation between the source and target domain is another group of robust speaker recognition. In this approach, a statistical or DNN-based compensation module performs a transformation between source and target speaker embeddings. In Section 3.5 several compensation techniques are reviewed.

In order to have a consistent notation, we consider the source and target environments. The source environment, $D^s = \{X^s = (X_1, ...X_n)\}$, is a situation where the baseline system is trained and the target environment, $D^t = \{X^t = (X_1, ...X_n)\}$, is a situation where the system will be used. It is assumed that $P(D^s) \neq P(D^t)$.

3.2. DATA AUGMENTATION

Data augmentation is creating new images of a speech signal by adding noise, reverberation, or changing speech characteristics such as speed. Data augmentation is an efficient strategy for robust speaker recognition. It has been used in almost all components of a speaker recognition system. Data augmentation techniques are used in two different ways. Firstly, they have been used directly in order to increase the size of training data or find a speech representation that is adapted to the target environment. Secondly, they have been used in preparing the training data for noise compensation techniques. In this section, the direct utilization of data augmentation techniques in speaker recognition systems is reviewed. The utilization of data augmentation with other methods will be explored in the next chapters.

The data augmentation techniques mostly are applied for environmental variabilities such as noise and reverberation. The impact of these variabilities is discussed in Section 1.4. In the case of reverberation, the RIR files can be recorded from real environments or they can be simulated [57]. For a given training dataset D^s using its new images, $D^{t_1}, ...D^{t_j}$, we can reduce the chance of overfitting which leads to achieving a more robust system in the presence of different variabilities [2, 58–60].

Data augmentation techniques are not limited to the signal level or adding noise and reverberation. They can be used at the feature level. Furthermore, they are not limited to adding reverberation and noise. Frequency masking, and time masking operations are used for data augmentation. SpecAugment is a speech data augmentation applied to filter bank features. This augmentation method is composed of time wrapping, frequency masking, and time masking operation on filter-bank features [61]. SpecAugment has been proposed in the speech recognition domain and it was investigated in TDNN and ResNet-based speaker embedding systems. It was

shown that with ResNet system it outperforms signal-level data augmentation [62].

WavAugment is an augmentation technique applied to unsupervised speech representation [63]. The proposed augmentation technique is an extension of the contrastive predictive coding speech representation (CDC) algorithm. The CDC produces the latent representation by predicting future samples [64]. In the WavAugment for a given speech segment, f, the latent representation is produced by predicting its past and future context where each of the past and future samples can use different images of augmented data. For example, it is possible to use different noises for the past and future samples [63]. In WavAugment, the next and previous frames are distorted with reverberation and additive noise in the time domain. This approach shows competitive performance in comparison to other data augmentation approaches such as SpechAugment [65].

Rawboost is a speech data augmentation method without using noise datasets or RIR files. It works by applying a combination of linear and non-linear convolutive noise, impulsive signal-dependent additive noise, and stationary signal-independent additive noise on raw speech signals. Rawboost can model different variabilities such as encoding, transmission, microphones, amplifiers, and both linear and non-linear distortions [65]. The Rawboost has been used in [66] with the Wav2Vec2 speech representation framework for robust speaker recognition.

The data augmentation hyperparameters such as SNR in the case of additive noise or the size of the room for reverberation are fixed or chosen randomly. This strategy doesn't guarantee the achievement of optimal diversified augmented data. In [67] a heuristic search algorithm is used in order to fine-tune the data augmentation hyperparameters. For a given speech signal it finds the probability and the intensity of applying a specific distortion such as additive noise and reverberation in signal level and masking parameters such as mask size at the feature level. The reported results show a significant improvement in comparison to the manual settings of data augmentation hyperparameters.

Since it is not easy to simulate all the possible variabilities in the speech signal, data augmentation can not be used as a universal methodology for various domain mismatches. This is the point that domain adaption techniques come into use. In the next section, the main domain adaption techniques used for robust speaker recognition are reviewed.

3.3. SPEECH ENHANCEMENT

Speech enhancement is a common way to remove the impact of external variabilities such as noise and reverberation in all speech applications. A general approach to suppress the negative impact of noise and reverberation in speaker recognition systems is by adding a speech enhancement module at the front end. The speech enhancement methods are applied at the signal level and feature level before the speaker modeling network. The general configuration of the speech enhancement method in speaker recognition systems is shown in Figure 3.1. The speech enhancement module can be optimized separately or jointly with the speaker modeling system. The feedback loop from the output of the speaker modeling part shows the cases of joint optimization. In joint optimization, the speaker classification output is used as guidance to help the training of the speech enhancement module.

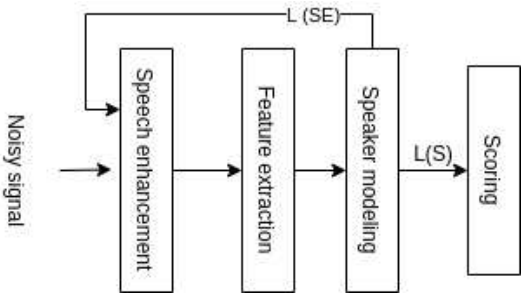


Figure 3.1: Speech enhancement module in speaker recognition systems

The output of speech enhancement methods generally is consumed by humans. Two main categories of speech enhancement metrics are called signal-level and perception-level [68]. Signal-level metrics are quantitative ways such as Source-to-distortion ratio (SDR), Source to interference ratio (SIR), and Source to artifact ratio (SAR) that measure the degree of enhancement or interference reduction [69]. Two main metrics in perception level are intelligibility and quality of speech signal [68]. In the case of speaker recognition systems, the main consumer of the speech enhancement methods is the speaker modeling module. Because of that reason, the speaker recognition metrics will be used as an evaluation metric and the speech enhancement metrics can be secondary metrics to explain the behavior of the systems after adding the speech enhancement module.

The speech enhancement techniques that have been used in the domain of speaker recognition can be categorized into three groups: mask-based, adversarial, and mapping-based techniques. The overall configuration of the three approaches is shown in Figure 3.2.

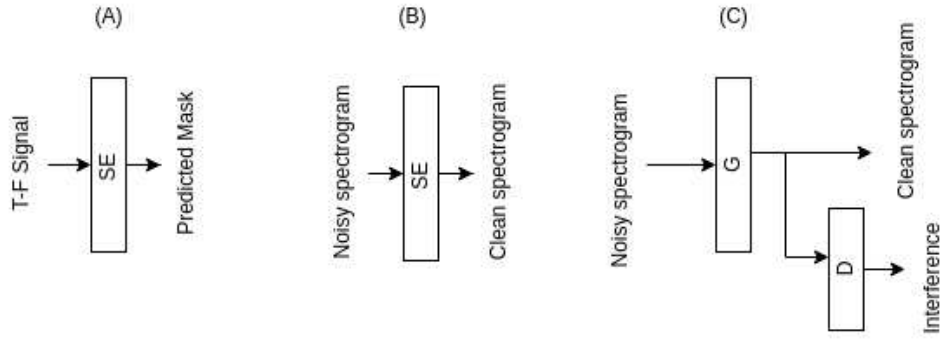


Figure 3.2: Speech enhancement approaches used in the domain of speaker recognition. (A) Masking-based approach; (B) Mapping-based approach; (C) GAN-based approach.

3.3.1. MASKING-BASED SPEECH ENHANCEMENT

In masking-based methods, the time-frequency relation between clean signal and interference is described [70]. Among the numerous masking-based methods Ideal Binary Mask (IBM) and Ideal Ratio Mask (IRM) are widely used in speaker recognition systems.

The IBM is defined as:

$$IBM = \begin{cases} 1, SNR(t,f) > LC \\ 0, otherwise \end{cases} \tag{3.1}$$

where t is time and f is frequency. For each unit, if SNR is bigger than a threshold $IBM = 1$, otherwise it is 0. IBM can be modeled as a binary classifier. Class 1 is for speech dominant units and class 0 stands for interference dominant units.

The IRM is defined as:

$$IRM = \left(\frac{S(t,f)^2}{S(t,f)^2 + N(t,f)^2} \right)^{\beta} \tag{3.2}$$

where $S(t,f)^2$ and $N(t,f)^2$ stand for speech and noise energy for a given unit, and β scales the mask that can be fine-tuned. In IRM the speech and noise are assumed to be uncorrelated. Mean square error (MSE) is the common loss function to estimate the IRM. The MSE is calculated between IRM_t, the targeted IRM and IRM_p the predicted IRM [71]. The masking-based speech enhancement is not limited to IBM and IRM. Other masking-based methods include:

- *target binary mask (TBM)* categorizes all T-F units with a binary label

- *spectral magnitude mask (SMM)* (or FFT-MASK) is the STFT (short-time Fourier transform) magnitudes of clean speech and noisy speech
- *phase-sensitive mask (PSM)* adds a measure of phase to the SMM
- *complex ideal ratio mask (cIRM)* reconstruct clean speech from noisy speech

The masking-based speech enhancement methods can be trained with the speaker modeling network jointly or separately. In joint optimization, it is possible to use both speaker and interference information. In [72] a loss function named VoiceID loss for speech enhancement was proposed that receives feedback from the speaker classifier. In this research, firstly a speaker modeling system is trained. In the second step, the speech enhancement network is added and the speaker modeling network is fixed. The speech enhancement network predicts a ratio mask at the output layer. The ratio mask is multiplied point-wise to a noisy signal in order to produce the enhanced signal and the speech enhancement network will be updated according to the speaker classifier loss value. The reported results show significant improvement of EER in the case of artificial noise on the Voxceleb1 test set. However, the Perceptual Evaluation of Speech Quality (PESQ) and Short Time Objective Intelligibility (STOI) metrics don't improve with the estimated mask, the performance of speaker recognition improves. Therefore, the speech enhancement method should be customized according to the speaker recognition task, however, an improvement in the quality of speech signal is not observed.

The second way of applying DNNs for masking-based speech enhancement is to train them separately. In [73] the researchers estimated the IRM mask with an LSTM-based speech enhancement DNN. They showed that obtaining high perception-based speech enhancement metrics doesn't guarantee good performance in downstream tasks such as speaker recognition and ASR. The proposed method is applied to artificial additive noise. A similar approach utilizes BLSTM and CNN speech enhancement networks to estimate IRM [74]. An encoder-decoder architecture based on convolutional gated linear units is proposed to estimate the IRM and PSM masks for the amelioration of magnitude and phase respectively [75].

A list of other masking-based speech enhancement methods is summarized in Table 3.1. The architecture stands for the DNN architecture, joint columns show the joint training of speech enhancement and speaker embedding networks, the interference column shows the type of noise (N) and reverberation (R) variabilities, and the speaker modeling network is the architecture of the speaker embedding extractor.

In our survey on mask-based speech enhancement techniques, we achieved the following main points:

Table 3.1: Masking based speech encasement.

Architecture	Mask	Joint	Interference	Speaker-modelling	Reference
TDNN	IRM	Y	N, R	x-vector	[72]
LSTM DAE	IRM	N	N, R	x-vector	[73]
BLSTM, CNN	IRM	N	N	i-vector	[74]
Gated CNN DAE	IRM, PSM	N	N, R	i-vector, x-vector	[75]
DAE	IRM	N	N	i-vector, GMM-UBM	[76]
BLSTM	WF, IRM GEV , MVDR	N	N, R	i-vector, x-vector	[77]

- The major part of the work is evaluated on artificial noise and reverberation and the generalizability is not shown for real cases [72, 75–77].
- The improvement is marginal and inconsistent. For example, the results are sensitive to the mask's hyper-parameters and the nature of speech such as SNR.
- The masks are sensitive to the type of interference. For example, IRM gives better results for additive noise due to amplitude improvement while PSM works better for phase improvement [75].
- The speech enhancement techniques need speaker information guidance to achieve better results. [72]
- The improvement of speech enhancement metrics doesn't guarantee the improvement of speaker recognition system [72, 73].

3.3.2. MAPPING-BASED SPEECH ENHANCEMENT

The target of mapping-based speech enhancement methods is to reconstruct the clean spectral features of a noisy speech signal [32]. In another term, mapping-based speech enhancement is suppressing the impact of noise at the feature level. Similar to masking-based speech enhancement methods, the mapping methods can be trained jointly with the speaker embedding network or they can be trained separately.

In [78] a UNet speech enhancement network is trained with an ECAPA-TDNN network. The speech enhancement network accepts STFT noisy features and reproduces the corresponding clean version by optimizing the MSE loss between denoised and clean features. The UNet network is trained jointly with the speaker embedding network. A similar approach is extended for noise and reverberation. In [79] the UNet speech enhancement module is trained by optimizing MSE loss, Fisher divergence loss, and speaker classification loss. The fisher divergence minimizes the

divergence between the gradient of the log density of the noise and the gradient predicted by the U-net speech enhancement network.

The separated training of mapping-based speech enhancement and speaker embedding extractor is addressed in several papers. In [80] an end-to-end speech enhancement method was proposed. In this work, a DAE based on a convolutional recurrent network (CRN) is used to estimate the clean version of the short-time Fourier transform (STFT) of clean speech. The speech enhancement network optimizes MSE between the noisy and clean versions of STFT. In [81, 82] a denoising autoencoder was proposed to map the noisy log magnitude spectrum to the corresponding clean version. The proposed DAE accepts a frame of noisy signal and its context, but the output layer reproduces the clean version of the central noisy frame, not its context. The denoised features perform well in both i-vector and speaker embedding frameworks for both artificial noise and reverberation but their generalizability is not tested in the case of real noise and reverberation.

Removing the impact of noise and reverberation on both amplitude and phase features can bring better results in speaker recognition systems. In [83] a speech enhancement DAE proposed that accepts MFCC as amplitude features and Modified group delay feature as phase features and reconstructs their clean version as a multitask regression problem. The speech enhancement network is trained for the reconstruction of both features jointly. The proposed approach shows that mapping both features gives better results than improving one of the amplitude and phase features. The main deficiency of reviewed literature is applying the proposed methods to artificial noise and reverberation.

3.3.3. ADVERSARIAL-BASED SPEECH ENHANCEMENT

Generative adversarial networks are another approach used in the front-end of speaker recognition systems. This approach is composed of a Generator (G) and a discriminator (D) [84]. The generator is an encoder-decoder-based architecture trained with a discriminator in an adversarial framework. The generator tries to recreate the clean version of the speech signal in a way to fool the discriminator (D).

In [85] a speech enhancement preprocessing is done for speaker recognition using the GAN architecture. In this paper, the architecture of G is U-Net that accepts the STFT spectrum, at the output of G the $L1$ distance between the clean signal and the predicted one is calculated as loss value, and the D tries to distinguish between the clean signal and the output of the G. In a similar approach [86], the researchers proposed a GAN feature bottleneck system composed of an Encoder (EN) and a Discriminator (D), the EN accepts the clean MFCCs and their corrupted versions with

a different noise, at the output it tries to predict a noise invariant version of MFCCs while D tries to classify the output of EN according to the type of noise.

3.4. ROBUST SPEAKER EMBEDDING TRAINING

Having robust speaker embedding is extensively explored in the phase of speaker embedding network training. At this level, the goal is to reduce the impact of variabilities during the training of the speaker embedding extractor. The main merit of this approach is eliminating the requirement of adding a specific module to compensate for the negative impact of variabilities.

3.4.1. ADVERSARIAL-BASED ROBUST SPEAKER EMBEDDINGS

GAN-based speaker embedding training is among the main approaches to achieve a nuisance robust speaker recognition system. A general framework for a GAN-based speaker embedding is shown in Figure 3.4.1. It can accept the data from both D_s and D_t domains. For a given speech signal the goal is to train the network to achieve nuisance invariant speaker embedding ω. For a given ω the classifier predicts the speaker's class $P(S|\omega)$ similar to a normal speaker embedding architecture described in Chapter 2, the loss value calculated by speaker classifier is L_{spk}. The discriminator network D is a feed-forward classifier network in general. The output of D depends on the type of nuisance. The GAN-based speaker embedding network optimizes the L_{spk} and L_d in an adversarial manner in a way that the discriminator can not distinguish between the different views of a speech utterance. The L_{GAN} is the summation of L_{spk} and the inverse of L_d.

By optimizing the L_{GAN} the discriminator tries to predict the domain of a given speech signal while the generator tries to produce embeddings that make the task harder for D. The adversarial speaker embedding learning aims to minimize the distance between the source and the target distribution feature spaces. This framework has been used to adapt the speaker embedding network for different variabilities.

- ***Additive noise:*** In [87] an adversarial strategy was proposed to make the speaker embeddings more robust against noise. In the standard speaker embedding extractors, after the embedding layer, a DNN speaker classifier is optimized. In this work, a second classifier is trained adversarially that classifies the type of noise in the output. In another work, a GAN-based speaker embedding was proposed that uses a binary discriminator to discriminate the noisiness of the speaker embedding alongside the speaker recognition classifier[88].

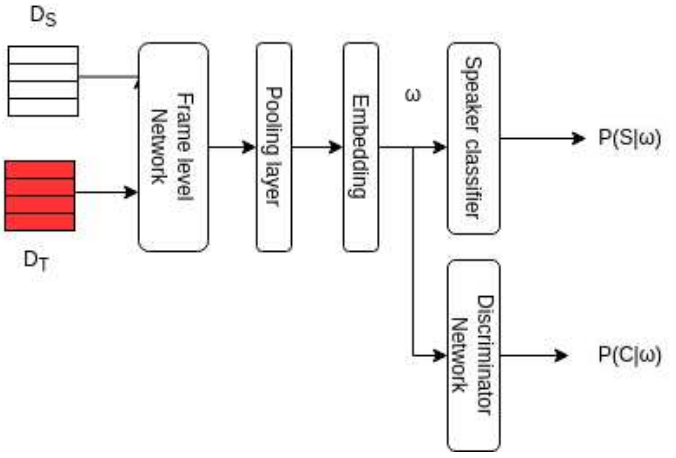


Figure 3.3: GAN-based robust speaker embedding training

- ***Channel mismatch***: Adversarial training is used successfully to produce channel invariant speaker embeddings [89]. In [90], two discriminators D_1 and D_2 fed with the output of the statistics pooling layer and speaker embedding layer respectively. The discriminator's output classifies different channels used in the dataset. It has been shown that the joint optimization of discriminators and speaker classifiers suppresses channel variability and improves performance significantly. A binary classifier is used in [91] as a discriminator, different views coming from the same speaker are considered as class 0, otherwise, the class is 1. The final goal is to give the same representation for the same utterances coming from different channels.
- ***Language Variability :*** In [92], the ResNet-based speaker embedding network was extended with a language discriminator. The discriminator is a binary classifier optimizing a cross-entropy, the class 0 is for the source and the target domain (i.e. in this case languages).
- ***Phonetic variability*** Phonetic information is another source of session variability that reduces the performance of text-independent speaker recognition systems. In [93] it is shown that suppressing the phoneme information at the segment level of DNN-based speaker recognition systems improves performance. In this research, the output of the discriminator is defined as the relative frequency of each phoneme presented in the input utterance.

3.4.2. DISCREPANCY-BASED DOMAIN ADAPTATION

Imposing a general and flexible constraint on target distribution with discrepancy-based methods during training/adapting the speaker embedding network is another popular domain adaptation approach. The main configuration of this approach is shown in Figure 3.4. For a given D_s source and D_t target distributions the discrepancy-based domain adaptation aligns the statistical distribution shift between the source and target domains using some measurements. The discrepancy constraint will be defined in the form of a regularization term L_{reg} and it will be optimized with the speaker classifier loss function L_{spk}.

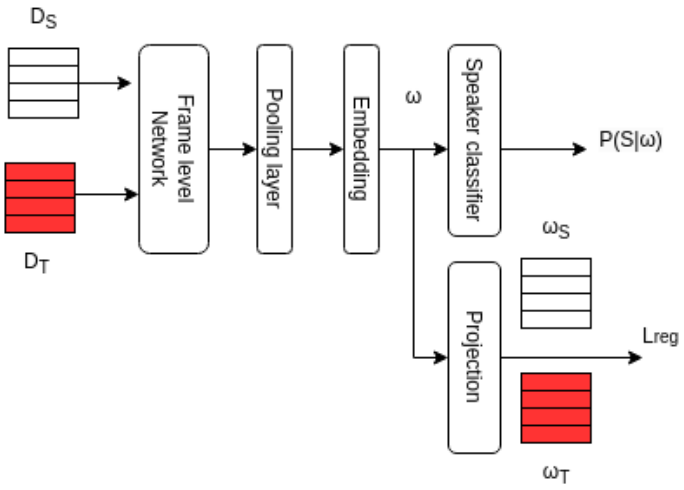


Figure 3.4: Discrepancy-based domain adaptation framework

The CORAL loss is the distance between the second-order statistics (covariance) of the source and target features [94] defined as:

$$L_{coral} = \frac{1}{4d^2}\|C_s - C_t\|_F^2 \tag{3.3}$$

where $\|.\|_F^2$ denotes the squared matrix Frobenius norm, d is the dimension of speaker embeddings, and C_s and C_t are covariance matrices for embeddings extracted from the source and target domains respectively. The CORAL discrepancy method has been used in [95] for speaker embedding domain adaption.

A well-known measure of discrepancy between domains is Maximum Mean Discrepancy (MMD), which is defined as:

$$L_{MMD} = \mathbb{E}[k(\Phi_f(x_S), \Phi_f(x_S^{'}))]_{x_S, x_S^{'} \sim D_S}$$

$$+ \text{ E } [\text{k}(\Phi_f(x_T), \Phi_f(x_T^{'}))]_{x_T, x_T^{'} \sim D_T}$$

$$\text{-2 E } [\text{k}(\Phi_f(x_S), \Phi_f(x_T^{'}))]_{x_S \sim D_S, x_T \sim D_T} \quad (3.4)$$

where $\Phi_f(x_S)$ and $\Phi_f(x_S^{'})$ are activation functions and k is Gaussian kernel defined as:

$$k(\Phi_f(x_S), \Phi_f(x_S^{'})) = exp(-\frac{\|\Phi_f(x_S) - \Phi_f(x_S^{'}\|_2^2}{2\sigma^2}) \quad (3.5)$$

In [96] MMD is used in optimization at both frame-level and utterance levels. During the training, the multilevel MMD is minimized jointly with the speaker classifier. This approach is compared with the CORAL discrepancy method and it reduces the impact of language variability to a significant degree. The MMD discrepancy method is used for channel adaptation in language recognition systems [97].

3.4.3. POOLING STRATEGY

A part of fine-tuning speaker embedding extractors for domain mismatch and handling variabilities is the pooling strategy. Statistics pooling (i.e. the concatenation of average and standard deviation) is among the mostly used pooling strategies. In [98] the performance of different pooling strategies with both TDNN and ResNet speaker embedding networks are explored. The results show the supremacy of using standard deviation pooling instead of statistics pooling with Voxceleb and SRE 2016 benchmarks. The same behavior was shown again for channel and language mismatch on SRE 2021 benchmark [99].

3.4.4. ATTENTION MECHANISM

Speaker embeddings encode information such as variabilities information over their dimensions. This secondary information is not essential for speaker recognition tasks. Guiding the training of the speaker embedding extractor towards a trajectory that gives bigger weight to parts of the speech signal that result in more speaker discriminant representation is called the attention mechanism.

In [100] the TDNN-based speaker recognition system (Figure 2.8) was modified by changing the statistics pooling with attention statistics pooling described in Section 2.4.5. Their experiments show a significant improvement in speaker recognition systems for noisy environments. A key reason behind the success of ECAPA-TDNN [5] and MFA-Conformer [4, 101] is leveraging the attention mechanism.

In [102, 103] a hierarchical attention mechanism is used at the frame level and segment level. For, $S = \{S_1, S_2, ..., S_N\}$, a given speech signal, composed of N segments and each segment $S_i = \{X_1, X_2, ..., X_M\}$ composed of M frames, the attention mechanism rescale each frame with softmax function as Equation 2.18 and 2.19. The same mechanism is repeated at the segment level. In [104] a speech enhancement DNN is integrated with the attention model. For an input, $X \in R^{T,F,C}$, where T, F, and C denote time, frequency, and channel dimensions, the attention mechanism is applied consequently across all dimensions of CNN blocks in the speech enhancement module.

3.5. SPEAKER EMBEDDING TRANSFORMATION

In this section, we review the speaker embedding transformation methods used at the speaker embedding level. A great part of the work done at this level is reconstructing the desired speaker embeddings, X_s, from their distorted corresponding version X_t by doing a transformation $X_t \mapsto X_s$. These methods are also known as noise compensation methods in the literature.

3.5.1. STATISTICAL SPEAKER EMBEDDING TRANSFORMATION

The statistical noise compensation modules are widely used and developed in the i-vector framework. In [105, 106] a linear transformation function is defined to reconstruct the X_s clean version of, X_t, noisy i-vector by a MAP estimator. The i-MAP method assumes that noisy and clean distributions are independent distributions, while in the reality it is not the case. The joint i-MAP is another statistical mapping technique that takes advantage of both clean and noisy i-vectors distributions along with the joint information between both of them [107]. In this approach three distributions are defined where X is the clean i-vectors, Y is the noisy i-vectors and Z is the concatenation of clean and noisy i-vectors that holds the joint distribution of clean and noisy i-vectors. The denoising problem is formulated with an MMSE (Minimum Mean Square Error) estimator. The i-MAP and joint i-MAP compensation functions give a significant improvement in the case of artificial noise and reverberation. The Correlation Alignment (CA) algorithm is used for domain adaptation in the speaker embedding (x-vector) space. The CA algorithm tries to minimize the distance between the covariance of the out-of-domain and in-domain speaker embeddings [108].

3.5.2. DNN-BASED SPEAKER EMBEDDING TRANSFORMATION

RECONSTRUCTION-BASED

The reconstruction-based methods at the signal level are called speech enhancement, which has been described in Section 3.3, the idea behind the mapping-based robust training of speaker embedding in Section 3.4 is the same, compensation techniques refer to reconstruction-based methods in speaker embedding level. The reconstruction methods are rigorous DNNs that try to reconstruct the clean version of noisy speaker embeddings. Denoising autoencoders (DAE) are among the common reconstruction DNNs.

The speaker discriminant denoising autoencoder (DDAE) proposed for noise compensation in i-vector framework [109–111]. The DDAE architecture is composed of a DAE and a feed-forward neural network (FFNN). The DAE is trained by minimizing a reconstruction loss between noisy and clean speaker embeddings in the input and output, the FFNN is trained by minimizing a speaker classification loss function. In [109] MSE is used as reconstruction loss and in [110] the cosine distance was minimized. The architecture of DDAE is depicted in Figure 3.5. In this approach, the classifier is jointly trained with a DAE in order to make the new i-vector more discriminant.

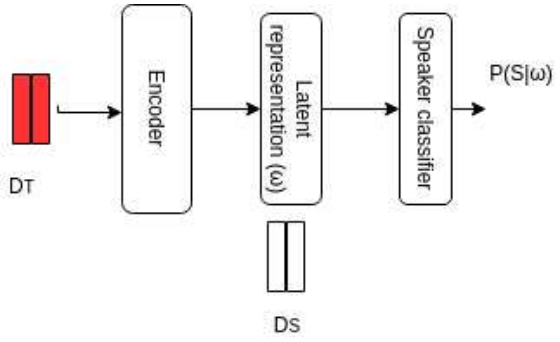


Figure 3.5: Discriminant DAE

Siamese speaker embedding reconstruction was introduced in [112] to compensate for additive noise and sampling rate mismatch in the speaker embedding (x-vector) framework. It is shown that this technique is more effective to compensate for sampling rate mismatch rather than additive noise. The Siamese network accepts pairs of noisy speaker embeddings $Y1$ and $Y2$ belonging to $S1$ and $S2$ speakers. The encoder network is trained by optimizing three objective functions. The $L1_{reconstruct}$ and $L2_{reconstruct}$ reconstruct the desired version of $Y1$ and $Y2$ in a mini-batch. The

third objective function is $L_{siamese}$ which is defined as:

$$L_{siamese} = \frac{f(Y1).f(Y2)}{\|f(Y1)\| \times \|f(Y2)\|} - \gamma(Y1, Y2) \tag{3.6}$$

where $\gamma(Y1, Y2)$ is:

$$\gamma(Y1, Y2) = \begin{cases} 1 \ where \ S1 = S2 \\ 0 \ otherwise \end{cases} \tag{3.7}$$

The architecture of the Siamese noise compensation network is shown in Figure 3.6.

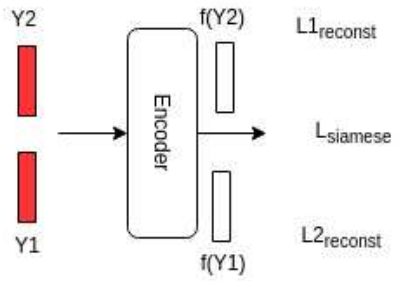


Figure 3.6: Siamese DAE for noise and channel mismatch compensation

DISCREPANCY

The MMD approach is used for the domain adoption method in the i-vector framework [113]. In this research, multiple auto-encoders were used to reconstruct input i-vectors coming from a specific domain. The MMD between hidden layers coming from different autoencoders is minimized to suppress the domain mismatch information from i-vectors [113, 114].

ADVERSARIAL TRAINING

GAN-based domain adaptation is another approach used in the i-vector framework [115]. The combination of variational auto-encoders and GANs is another approach for noise compensation at the speaker embedding level. In [116] a variational auto-encoder is integrated with an adversarial domain discriminator and a speaker classifier. The architecture of the proposed systems is shown in Figure 3.7. The encoder receives speaker embeddings from different domains. The latent space is fed into the speaker classifier, domain classifier, and decoder. The adapted version achieves a marginal improvement on SRE16 and SER18 evaluation benchmarks.

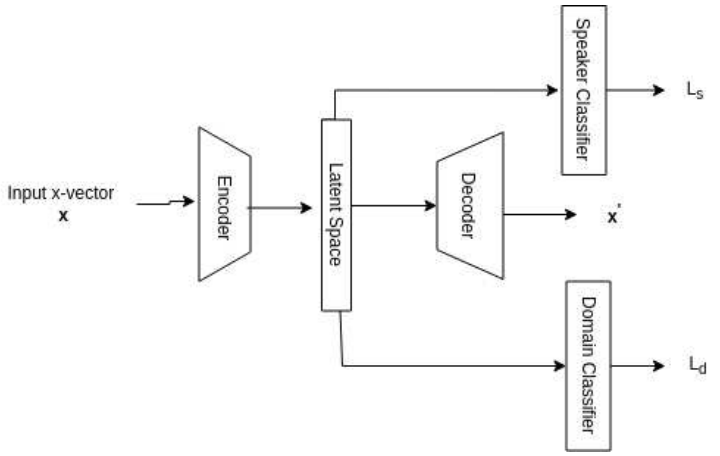


Figure 3.7: Variational GAN variability compensation

Cycle-GAN is another framework that is used for domain adaptation in the x-vector framework. Cycle-GAN is a GAN framework that is composed of $G : X \rightarrow Y$, $F : Y \rightarrow X$ generators, and D_X, D_Y discriminators. D_Y impose on G to transform X into outputs indistinguishable from domain Y, and vice versa for D_X, F, and X. Two cycle consistency losses insure that the transformations between source and target domains are reversible: (b) forward cycle-consistency loss: $x \rightarrow G(x) \rightarrow F(G(x)) \approx x$, and (c) backward cycle-consistency loss: $y \rightarrow F(y) \rightarrow G(F(y)) \approx y$. In [117] the cycle-GAN framework is used for domain adaptation between microphone and telephone speech in the SITW benchmark. In another research, this framework is used to do a transformation between x-vectors in a noisy and clean environment. They achieve a significant improvement of EER in the VoiCes benchmark [118]

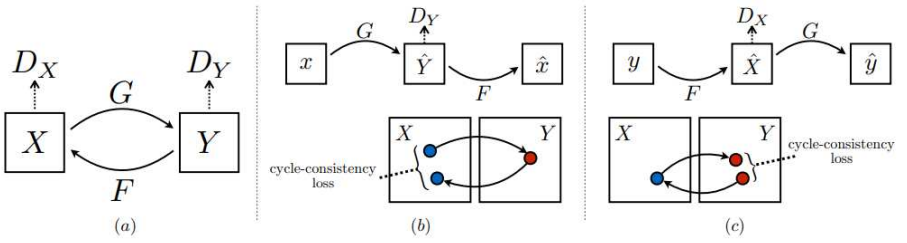


Figure 3.8: Cycle-GAN framework for speaker embedding adaptation [119]

3.6. CONCLUSION

In this chapter, the main robust speaker recognition approaches are reviewed. Firstly the data augmentation techniques at different levels were reviewed. However data augmentation improves the performance of speaker recognition systems in general by increasing the number of speakers and data diversification, still, we need adaptation methods that directly target a specific variability.

In this chapter the categories of speech enhancement methods including masking-based, mapping-based, and adversarial training approaches reviewed that are extensively used in the speaker recognition domain. The main limitation of masking-based approaches is their sensitivity to the type of interference. Also, the improved speech signal quality doesn't guarantee to improve speaker recognition system. The majority of mapping-based and adversarial training methods are tested with artificial noise and reverberation. The reviewed literature shows that joint training of speech enhancement module and speaker embedding network is more promising than separate training. In the case of joint training, the speech enhancement network uses the feedback given by the speaker embedding network.

Another common approach is training a robust speaker embeddings extractor. This approach can be applied to a bigger number of variabilities in comparison to data augmentation or speech enhancement. Having a robust embedding extractor can get rid of us from using extra subsystems such as a speech enhancement module.

Doing a transformation between source and target speaker embedding before scoring is another group of robust speaker recognition approaches reviewed in this chapter. Different DNN compensation modules such as DAE, GAN, or VAE architectures are used for domain adaption at the speaker embedding level. The reviewed literature shows that noise compensation at the speaker embedding level is more promising in comparison to other approaches. Using compensation techniques we don't need to change the architecture of the speaker embedding extractor which leads to less computational cost and increases the adaptability of the trained speaker embedding extractor.

4

DATASETS AND BENCHMARKS

If machine learning was something you bought in the supermarket, its carton would say: Just add data!

Pedro Domingos

In this chapter, the main speaker recognition datasets and evaluation benchmarks are reviewed. Among the training datasets, the Voxceleb 1&2 corpus is reviewed. All datasets used in the contribution part of this thesis are described in this chapter. The MUSAN, BBC Noise, and Freesound noises are datasets that are used for data augmentation or data simulation. The third groups of datasets are evaluation benchmarks. Among the evaluation benchmarks we reviewed SITW, VoxSrc, NIST SRE 2021, ChiMe, VoiCes, and Dipco datasets. The last part of this chapter is devoted to the RoboVox dataset, which we introduce for the first time to the community.

4.1. INTRODUCTION

The datasets used in the domain of speaker recognition can be categorized into three groups. The first group is speaker embedding extractor training datasets. According to our discussion on the general framework of DNN-based speaker recognition, the main annotation for this kind of dataset is the speaker label. However, having extra information can help in designing specific speaker embedding extractors. Furthermore, having a significantly big size in terms of the number of utterances and speakers is another requirement for training datasets. The common speaker embedding training datasets are discussed in Section 4.2.

The second category of datasets used in speaker recognition is data augmentation datasets. Data augmentation datasets can be used for several purposes. Above all, they can be used during training the speaker embedding because of two main reasons. Firstly, using data augmentation techniques we can increase the size of speaker-annotated datasets that are crucial for training large DNNs with millions or even billions of parameters. Secondly, data augmentation increases the variability of the dataset. This technique makes the speaker recognition systems more robust against unwanted interferences such as noise and reverberation. Having a piece of prior information about the target environment can help us to do data augmentation in a manner that matches the interference. Also, data augmentation can be used in training noise compensation techniques. In Section 4.3 three main datasets that are used in this thesis are described.

The third group of required datasets is evaluation benchmarks. The speech files in this group can be divided into enrollment and test files. The enrollment files are used for registered speakers and test files are claimed users that should be authenticated by the speaker recognition systems. Depending on the application, different kinds of annotations are required. For example, in text-dependent speaker recognition systems, transcription is required. In our case, text-independent speaker recognition systems, we just need the speaker label. For the semi-supervised or weakly-supervised adaption technique, a small number of files in the evaluation dataset can be used to adapt the system to the target environment. Several benchmarks and standard evaluation datasets are described in Section 4.4.

4.2. SPEAKER EMBEDDING TRAINING DATA

Voxceleb Corpus has been collected in two stages: Voxceleb1 and Voxceleb2. Voxceleb1 contains 100,000 utterances from 1,251 celebrities. Voxceleb2 includes over 1 million utterances from 6,000 speakers. The speakers are from different ethnicities with

different ages, professions, and accents. The utterances are degraded with different types of noises including background chatter, laughter, overlapping speech, and room acoustics. Although there are a lot of variations in recording devices and channels. We used Voxceleb1 and Voxceleb2 in training the speaker embedding extractor networks and training the denoising techniques [120]. The general specification of the Voxceleb dataset is presented in Table 4.1.

Table 4.1: Dataset statistics for both VoxCeleb1 and VoxCeleb2

Dataset	Voxceleb 1	Voxceleb 2
No^o of speakers	1251	6112
No^o of utterances	153,516	1,128,246
Avg no^o of utterances per speaker	116	185
Avg length of utterances (second)	8.2	7.8

Among other useful datasets for training speaker recognition systems, we can name Common Voice 12.0 which its English part contains 3,161 hours of validated speech recorded from 85,825 unique utterances [1].

4.3. DATA AUGMENTATION DATASETS

4.3.1. MUSAN

The Musan corpus has 109 hours of voice data including 60 hours of speech, 42 hours of music, and 929 noise files. The noise files include DTMF tones, dial tones, fax machine noises, and ambient sounds, such as car idling, thunder, wind, footsteps, paper rustling, rain, animal noises, etc. Also, it includes some recordings of crowd noises with indistinct voices [121].

4.3.2. FREESOUND

This corpus included 3000 RIR files and 4275 noise files collected from Freesound noises. The selected noise categories include door, keyboard, office, phone, background noise in the room, printer, fan, door knock, babble, etc. We divided the dataset into two sets: a training set composed of 3725 clips and an evaluation set composed of 1000 clips [122].

[1] https://commonvoice.mozilla.org/en/datasets

4.3.3. BBC NOISES

The BBC noises dataset includes 33,000 sound clips across the world from the past 100 years. This dataset includes clips made by the BBC Radiophonic workshop, recordings from the Blitz in London, special effects made for BBC TV and Radio productions, as well as 15,000 recordings from the Natural History Unit archive[2].

4.4. EVALUATION BENCHMARKS

4.4.1. VOXCELEB

The Voxceleb providers defined several benchmarks on the Voxceleb data. VoxCeleb1-E is a test protocol that covers all speakers in Voxceleb1. The good results with the state-of-the-art robust speaker embeddings on Voxceleb protocol stir up the necessity of new benchmarks for more severe environments [9]. VoxSrc is the Voxceleb speaker recognition challenge that each year defines new test sets and tasks in more challenging situations [7].

4.4.2. SITW

The Speakers in the Wild (SITW) speaker recognition database contains hand-annotated speech samples from open-source media to benchmark text-independent speaker recognition technology on single and multi-speaker audio acquired across unconstrained or wild conditions. The database consists of recordings from 299 speakers, with an average of eight different sessions per person. Unlike existing databases for speaker recognition, this data was not collected under controlled conditions and thus contains real noise, reverberation, intra-speaker variability, and compression artifacts.

The SITW is among the earliest known speaker recognition challenges. While at the beginning of DNN-based speaker recognition systems, this dataset was challenging, similar to the Voxceleb test set, it is not challenging for the recent speaker recognition systems anymore[123]. In the following, more challenging benchmarks are discussed.

4.4.3. NIST

The Speaker Recognition Evaluation (SRE) is a global benchmark held by the National Institute of Standards and Technology (NIST) since 1996. The latest version of the NIST 2021 protocol is shown in Table 4.2. The speech files are received from the

[2] http://bbcsfx.acropolis.org.uk

WeCanTalk corpus. This corpus is composed of multilingual phone calls and video recordings collected from social media [8]. This feature makes the corpus suitable for a channel or microphone mismatch condition.

Table 4.2: NIST evaluation benchmark specifications

Task	Spks(M/F)	Enroll	Test	Target	Non-target
DEV	5/15	138	2001	14,458	177,793
TEST	43/139	1247	17,037	132,038	5,899,731

4.4.4. VOICES

The VoiCes is a replayed speech corpus recorded from Librispeech under different types of noises in four rooms. The VoiCes corpus takes several variables into account that makes the impact of noise and reverberation severe. Among those variables angle, distance, and shape of the room are more significant. In VoiCes, there is an L-shaped room. Since in such environments, there is no direct path from the speaker to the microphone the impact of reverberation becomes more serious. Also, the angle between the microphone and speaker is variable. Similar to the shape of the room, the indirect path between the microphone and speaker makes the impact of reverberation more severe. Besides these positive aspects of VoiCes, there are two main deficiencies. Firstly it is replayed speech and there is convolution noise of the loudspeaker that makes the situation far from real applications. Also, the speech files are recorded from a small pool set of clean speech (i.e. 2583 files from Librispeech) that reduce the content variation in the data [6].

4.4.5. DIPCO

The Dinner Party Corpus (DiPCo) is a speech database that replicates the scenario where a group of people is having an interactive conversation while having dinner in a simulated home environment. The DiPCo speaker recognition benchmark is derived from the DiPCo corpus. In this benchmark three tasks are defined: 1) Far-field speaker recognition from a single microphone; 2) Task Far-field speaker recognition from a single microphone array; and 3) Far-field speaker recognition from distributed microphone. The overall specification of the DiPCo benchmark is presented in Table 4.3. $Trials$ refers to the number of evaluation target/nontarget pairs, $Target$ refers to the number of evaluation trials positive, $Utterances$ refers to the total number of unique speech segments in the test set, and, $Duration$ is the length of files reported as min/mean/max [9].

Table 4.3: DiPCo speaker recognition benchmark specifications

Task	**Trials**	**Target**	**Utterances**	**Duration(s)**
Task-1	900,000	150,000	4,405	3.51/5.03/7.99
Task-2	900,000	150,000	4,405	3.51/5.03/7.99
Task-3	900,000	150,000	30,835	3.51/5.03/7.99

4.4.6. CHIME

This speaker recognition benchmark is derived from the CHiME-5 corpus. The goal of this benchmark is to foster research on far-field multi-speaker recordings of naturally occurring spoken interactions. This corpus includes four tasks with single-speaker and/or multi-speaker recordings. The tasks are defined to compare close-talking vs distant/far-field microphone recordings and single-microphone vs microphone-array situations. Their evaluation shows that in the case of multispeaker, the EER is equal to 20 which means for difficult cases we are from an ideal speaker recognition system. However, the reported results make this corpus a good candidate to improve the performance of speaker recognition, unfortunately, the dataset is not currently available for public use [124].

4.4.7. FABIOLE

Fabiole is the most used corpus in the experimental part of this thesis. Fabiole is a French corpus that contains 7,000 files from 130 speakers. The length of files spans from very short utterances of less than 2 seconds to very long utterances. The Faboile corpus is used as a test and enrollment. In this corpus, there are 30 speakers with more than 100 utterances. The remaining 100 speakers have only one session. The motivation behind this design is to have enough target trials for enrollment speakers and use the 100 speakers with one file to create non-target trials [125]. However, in our experiments, we don't respect this implication in some cases in order to make a severe and difficult situation for our speaker recognition system. The details about different protocols created from this corpus are described in the next chapters.

4.5. ROBVOX DATASET

In this thesis, a new corpus (named RoboVox) will be introduced to the community. Robovox is a French corpus recorded by a mobile robot (E4) in the framework of the ANR project RoboVox. The robot is equipped with a speech recognition system in noisy environments (Fig 4.1).

Each recording in this corpus has 6 channels. The fifth channel is a closed microphone which we considered as clean and other channels are considered noisy and reverberated[3]. The utterances are recorded in both open and closed spaces. The distance between the speaker and microphones in far channels is between 1 and 3 meters. There is a sixth microphone that records the robot's background noise.

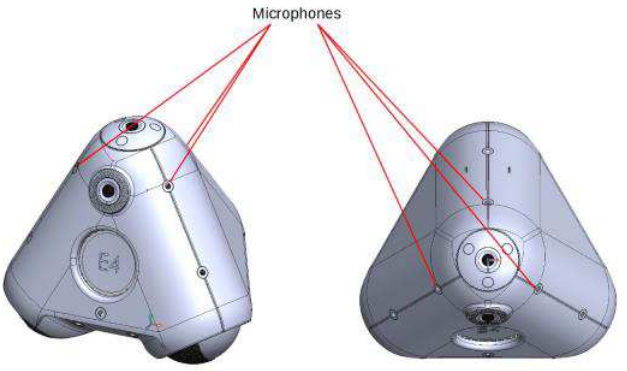


Figure 4.1: Robovox (E4): a mobile robot

The recorded part includes 2219 conversations from 64 speakers including 30 males and 34 females. In each conversation, there are 5 dialogues on average. Therefore, the total number of recorded dialogues is $\simeq 11,000$. The average length of each dialog is 3.6 seconds. The final version will include 70 speakers with 11,000 dialogues (speaker turns) between the robot and the speakers.

Each recording has 6 channels. The channels information is as follows:

- **Channel 1 to 3:** microphones on the angels of the robot
- **Channel 4:** microphone embedded inside the robot
- **Channel 5:** ground truth microphone which is close to the speaker
- **Channel 6:** voice of the robot

It deserved to be mentioned that the clean signal, recorded by channel 5, is a positive point that makes the RoboVox dataset protocol more comprehensive in comparison to the previous far-field speaker recognition datasets. This feature enables us to have the best-expected baseline system and allows us to know the amount of performance degradation for far-field microphones.

The files are recorded from different distances in different acoustical environments

[3]https://robovox.univ-avignon.fr/

with the main following settings:

- **1m, 2m and 3m:** Distance of the speaker from the robot, respectively 1, 2, and 3 meters.
- **hall, open space, small room (open/close) and medium room (open/close):** The sessions are recorded in the different rooms/environments with the door open or close in meeting rooms.
- **wall, center and corner:** The robot is placed close to a wall (or window), in the center of the room, or in the corner respectively.
- **calm or noisy:** Level of noise in the environment.

4.6. CONCLUSION

In this chapter, we introduced the main training corpora, data augmentation corpora, and evaluation benchmarks. Among the introduced datasets we use Voxceleb1 and Voxceleb2 broadly in this thesis for two purposes: training DNN speaker embeddings and training DNN-based noise compensation techniques. The MUSAN, BBC, and Freesound noises are used in different tasks in the next chapters. Among the evaluation benchmarks, we used the Fabiole, Voices, and RoboVox datasets during our experiments.

5

NOISE COMPENSATION IN SPEAKER EMBEDDING LEVEL

In this chapter, we propose a framework for noise compensation at the speaker embedding level. Indeed the proposed framework is a pipeline that adds a noise compensation module before scoring. We propose several denoising techniques. Firstly, we use the i-MAP method which considers that both noise and clean speaker embeddings have a Gaussian distribution. Then, leveraging denoising autoencoders (DAE) we try to reconstruct the clean speaker embedding from the corrupted version. After that, we propose two hybrid systems composed of statistical i-MAP and DAE. Finally, we propose a novel DAE architecture, named Deep Stacked DAE. In the first experimental part, the proposed noise compensation techniques are used for additive noise compensation. The results obtained by Deep Stacked DAE are almost better than other methods. For utterances longer than 12 seconds, we achieved a 51% relative improvement in terms of EER with Deep Stacked DAE, and 18% for utterances shorter than 2 seconds compared to the baseline system. The specific noise compensation, and exploring the joint data augmentation and noise compensation are other experiments done in this chapter. We will show even in the case of having extensive data augmentation, the noise compensation can bring us closer to a clean environment. In the last part of this chapter, we explored the scenarios where there are several distortions including additive noise and reverberation. We will propose two configurations for the presence of multiple distortions.

The presented work in this chapter is published in [126], [53] [127]

5.1. INTRODUCTION

In the past decade, introducing the i-vector statistical model and x-vector[1] speaker embedding has led to notable progress in the speaker recognition area. However, the x-vector speaker modeling method has caused substantial improvement in the speaker recognition system, the performance of this system in challenging environments with the presence of unseen noises and reverberation degrade significantly. In our experiments, with low SNR and a large number of unseen noises added to the test data, we observed that the performance of x-vector embedding diminishes drastically in comparison to results obtained with noise-free x-vectors.

Some studies [58–60] propose that by increasing the number of speakers, the amount of training data, and by use of data augmentation, the x-vectors can achieve a certain degree of robustness with the presence of noise, but this degree of robustness remains insufficient, especially in the case of low SNR.

In this chapter, in addition to the common data augmentation techniques, we propose to denoise the noisy x-vectors to approach the performance obtained by noise-free x-vectors. In this manner, it becomes easier to target a specific unseen noise or to adapt the denoising system to new conditions. In other words, the proposed system is a pipeline of two systems, the first allows generating the best x-vectors possible, and the second is used to denoise x-vectors.

A major part of the robust speaker recognition research against noise and reverberation has been done at the front end: signal level, feature level, and speaker modeling level (i.e. speaker embedding extractor). In the first part of our work, we want to focus on noise compensation at the speaker embedding level (i.e. the speaker embeddings obtained by DNNs). Thanks to its statistical properties, developing denoising techniques at the speaker embedding level is more promising and easier than the signal level, feature level, or speaker modeling network. However, the noise compensation at the speaker embedding level has been done successfully and extensively in the i-vector space, there is no such exploration in the x-vector systems. The success of this methodology motivated us to extend it in the framework of x-vector systems. The previous research in the i-vector framework shows that doing noise compensation at this level is more promising than speech enhancement at the frontend [74] [106]. Last but not least motivation is time and computational cost of noise compensation at the backend. Simple statistical transformations such as i-MAP need a small number of training examples and it can be done in a very short time while training a robust speaker embedding extractor takes several days.

[1] Throughout this chapter we use x-vector and speaker embedding interchangeably because we are using TDNN network for speaker embedding extraction

Firstly we describe the proposed framework for noise compensation at the x-vector level. After that, we introduce several noise compensation techniques. First of all, we apply statistical denoising techniques on x-vectors that work effectively in the i-vector domain. Our first attempt consists of using a statistical approach based on the use of maximum a posteriori (MAP), namely, the i-MAP approach [105]. The i-MAP denoising technique has been used successfully in the i-vector space. Then, we compare the results obtained by i-MAP with the denoising autoencoder. Furthermore, we propose two hybrid systems that use both denoising autoencoders and i-MAP. Finally, we propose a novel DNN, named Deep Stacked DAE that outperforms all the other methods.

The first part of our experiments is devoted to compensating additive noise. In this group of experiments, our goal is to develop a system that tries to compensate for different kinds of noises without explicit information about the noise for a given utterance in the test data. The robustness of all the proposed denoising techniques against additive noise is explored.

In the second part of our investigation, we compare the impact of data augmentation versus noise compensation. We will show that, despite having efficient data augmentation strategies, the noise compensation module brings a significant improvement to the speaker recognition system.

From the first and second parts of our experiments, we found the superiority of our proposed Deep Stacked DAE in noise compensation. In the next experiments, we just use this approach. In another experiment, we explored using a specific model for a known type of noise and we compare the results with the general model.

In many cases, there is more than one distortion. For example, the impact of additive noise in closed environments becomes more severe by reverberation. Proposing an efficient strategy for such cases is the last situation studied in this chapter. We will propose two different configurations for environments where there are additive noise, early reverberation, and late reverberation.

5.2. NOISE COMPENSATION FRAMEWORK

In this section, the framework of noise compensation at the x-vector level is presented. The proposed framework for noise compensation in speaker embedding level is composed of three steps. In the first step, the speaker embedding network is trained with extracted features from a clean dataset and augmented data. Additive noise and reverberation pools can be used to create several distorted versions of the clean speech signal. This trained network is used as a speaker embedding extractor in the

second part of the proposed pipeline (Figure 5.1. A).

In the second step, the trained speaker embedding extractor is used to produce training data for the noise compensation module. The noise compensation training data is composed of pairs of distorted/clean speaker embeddings. For generating the distorted version of the training data we can do it blindly or take the environmental information into account. After the extraction of both noisy and clean pairs of x-vectors, the noisy version should be passed through a noise compensation module. The goal is to train the noise compensation module in order to do a mapping between the noisy and clean pairs. It should be mentioned that the noise pool and the RIRs in the preparation of noise compensation training data are different from those used in training the speaker embedding extractor (Figure 5.1. B).

The trained noise compensation module is used in the third step to remove the impact of noise on speaker embeddings extracted over the test data. However, the goal is to generalize the trained noise compensation module for test files degraded with real distortions, in the experimental setup it is normal to simulate different situations with artificial noises. In this chapter, we focus on artificial noise and reverberation. The real noise and reverberation will be discussed in the next chapter. This process is depicted in (Figure 5.1. C).

There are two main tasks in this framework that can be explored to achieve better performance. The first task is simulating the training data for the noise compensation module. Seeking an efficient simulation method that models the real environment and makes the noise compensation generalizable to the real environment is a crucial task in this framework. We will discuss this issue in more detail in the next chapter. The second open question is designing an efficient noise compensation module. Our work in this chapter is mainly focused on this task. The proposed noise compensation modules are discussed in the next sections and their evaluation for different noises and reverberation comes after.

5.2.1. SPEAKER EMBEDDING EXTRACTOR

In this chapter, we perform noise compensation on the TDNN speaker embedding network. The TDNN architecture was introduced in chapter 2. In our experiments, we are using the TDNN network implemented in [2]. The speaker embedding network is composed of five frame-level TDNN layers. The context of TDNN layers increases in deeper layers. For a specific frame, t, the first layer accepts 5 frames in its context. This context will increase to 9 frames in the second layer and for the next hidden layer, the context captures 15 frames. The statistical pooling layers calculate the standard deviation and average for all T frames in a given utterance. The next layers

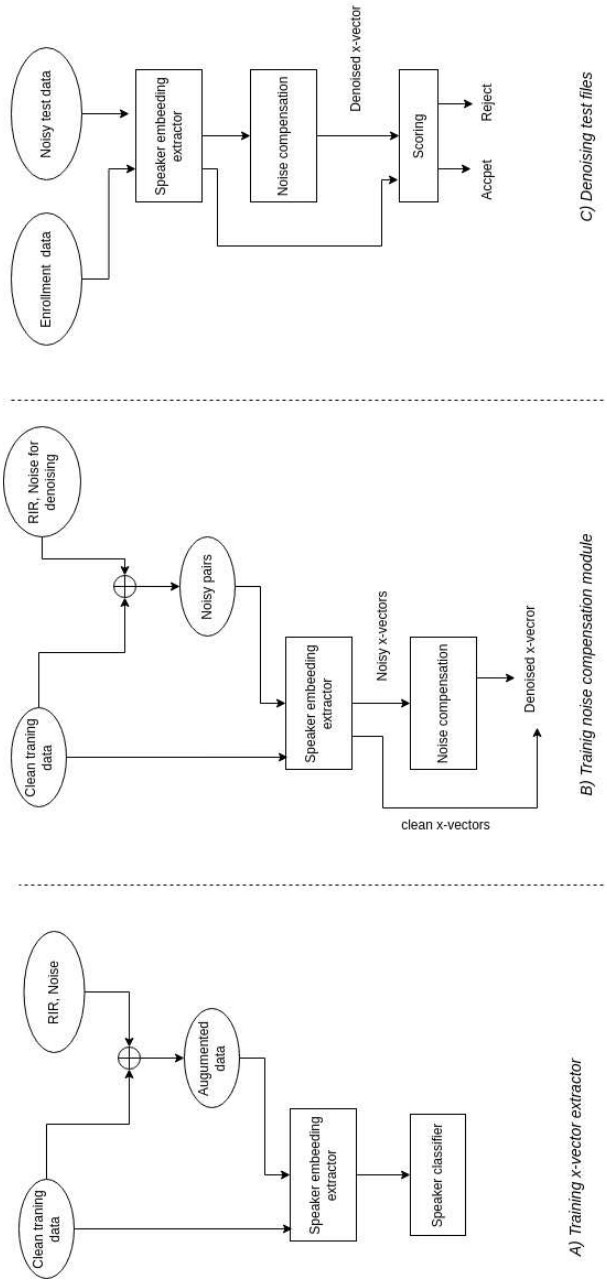


Figure 5.1: The proposed framework of noise compensation at speaker embedding level

are fully connected layers that work on the segment level. The last layer is a softmax layer that gives the probability of a specific utterance belonging to 1...N speakers in the training dataset. The architecture of the speaker embedding network is shown in Table 5.1. The network optimizes the Additive Angular Margin Loss function.

Table 5.1: The TDNN speaker embedding (x-vector) extractor

Layer	**Layer context**	**Total context**	**Input x output**
frame 1	[t - 2, t + 2]	5	120x512
frame 2	{t - 2 , t, t + 2}	9	1536x512
frame 3	{t - 3, t, t + 3}	15	1536x512
frame 4	{t}	15	512x512
frame 5	{t}	15	512x1500
stats pooling	[0, T]	T	1500Tx3000
segment6	0	T	3000x512
segment7	0	T	512x512
softmax	0	T	512xN

5.2.2. PROPOSED NOISE COMPENSATION MODULES

In this section, the details about the i-MAP, denoising autoencoders, and other proposed variants of DAE denoisers are discussed. Firstly, the i-MAP method is described. Then, denoising autoencoders are described and two hybrid systems developed from i-MAP and DAEs are presented. Finally, a novel DAE architecture is introduced.

5.2.3. I-MAP

The i-MAP is a statistical denoising method originally developed for the i-vector framework [105]. We define X and Y given random variables for clean and noisy x-vectors. A third random variable for noise is defined as:

$$N = Y - X \tag{5.1}$$

Where:

$$X \sim \mathcal{N}(\mu_X, \Sigma_X)$$

$$\text{N} \sim \text{N}(\mu_N, \Sigma_N) \tag{5.2}$$

In the i-MAP approach, we assume that both noise and clean x-vectors have a Gaussian distribution. For a given noisy x-vector, Y_0, from Equation 5.1 and Equation 5.2.3, we define:

$$f(Y_0|X) = \frac{1}{(2\pi)^{\frac{p}{2}}|\Sigma^{\frac{1}{2}}|} \exp^{-\frac{1}{2}(Y_0-X-\mu_X)^T\Sigma_N^{-1}(Y_0-X-\mu_X)} \tag{5.3}$$

To find X_0, the clean version of Y_0, a MAP estimator is used:

$$X_0 = argmax_x\{lnf(Y_0|X)f(X)\} \tag{5.4}$$

The final expression of calculating denoised x-vectors is:

$$X_0 = (\Sigma_N^{-1} + \Sigma_X^{-1})^{-1}(\Sigma_N^{-1}(Y_0 - \mu_N) + \Sigma_X^{-1}\mu_X) \tag{5.5}$$

More details about the mathematics of i-MAP can be found in [105, 106]

5.2.4. DENOISING AUTOENCODER

Denoising autoencoder is a specific type of autoencoder that takes the noisy x-vector as input and tries to reconstruct the clean version at the output. Denoising autoencoder tries to minimize:

$$L(x, f(y)) \tag{5.6}$$

where L is the loss function, y is the corrupted x-vector and $f(y)$ is the output of DAE [44]. In the simplest version, L is the MSE loss. We did several experiments to find the best denoising autoencoder. In the best architecture we found, three layers are used. The activation function of the input and output layers is linear. In the hidden layer, there are 1024 neurons with Tanh activation functions. The number of hidden layer neurons is twice the number of x-vector dimensions in order to avoid information loss. The [MSE) between the noisy and clean x-vector vector is calculated at each mini-batch (Equation. 5.7):

$$L_{MSE} = \sum_{i=1}^{B} \|x_i - y_i\|^2 \tag{5.7}$$

where B is the batch size, x_j is the jth clean x-vector and y_i is ith noisy x-vector.

5.2.5. HYBRID SYSTEMS

Here, we try to benefit from the potential of i-MAP and DAE simultaneously. In order to do this, we combine those systems in two ways:

DAE i-MAP: In this system, we used DAE and i-MAP methods sequentially. Firstly, the noisy x-vector is denoised by leveraging DAE, then the output of the DAE is denoised with i-MAP. The architecture of this system is presented in Figure 5.2 . A.

Gaussian DAE: Adding a regularization term to the loss function is a common way to constrain the solution space in denoising DNNs [128] [129]. In Bayesian formulation, the regularization terms correspond to prior probabilities added to the loss function. Similar to i-MAP, in Gaussian DAE, we want to impose on the output of DAE and the estimated noise to have a Gaussian distribution.

In the same manner, we add the regularization term to the MSE loss function. The proposed system is named Gaussian DAE. In Gaussian DAE we defined a new loss function to reduce MSE between input and output, and simultaneously maximize the a priori probabilities of the noise and of the obtained x-vectors (both assumed to be Gaussian) like in the i-MAP. In this system, the following equation is used as a Gaussian regularizer:

$$L_{reg} = (Y - X - \mu_N)\Sigma_N^{-1}(Y - X - \mu_N) + (\text{X-}\mu_X)\Sigma_X^{-1}(X - \mu_X) \tag{5.8}$$

where Y is the input of DAE and X is the output of DAE at each mini-batch, μ_N and μ_X are the means of noise and clean x-vectors respectively, Σ_N^{-1} and Σ_X^{-1} are the inverse of the covariance matrix for noise and clean data, all these parameters are estimated on the whole training data. The Gaussian DAE minimizes the following equation:

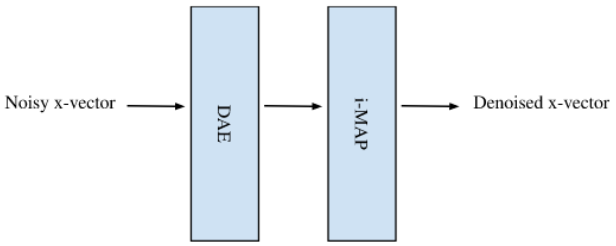
$$L_{Gaussian} = L_{MSE} + \lambda L_{reg} \tag{5.9}$$

The architecture of the hybrid systems is shown in Figure 5.2. B. In both cases, the architecture of DAE is the same as Section 5.2.4.

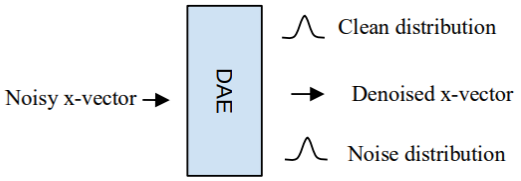
5.2.6. DEEP STACKED DENOISING AUTOENCODER

In this subsection, we introduce a new denoising autoencoder called Deep Stacked DAE. In this architecture we have several DAE blocks. The noisy x-vectors, Y, fed to the first DAE. The next DAE block receives X_i, the output of its predecessor block i concatenated with $Y - X_i$, the difference between noisy x-vectors, and the output of the previous block. The stacked DAEs are trained jointly with the SGD optimization algorithm. The architecture of Deep Stacked DAE is presented in Figure 5.3. The architecture of DAE blocks is the same as DAE presented in Section 5.2.4.

As we will see in the next section, this architecture outperforms all other methods. One assumption behind the effectiveness of this method is that using the difference



(A) the hybrid DAE i-MAP



(B) Gaussian DAE: denoised x-vectors have a Gaussian distribution.

Figure 5.2: The hybrid noise compensation module composed of statistical i-MAP and DAE

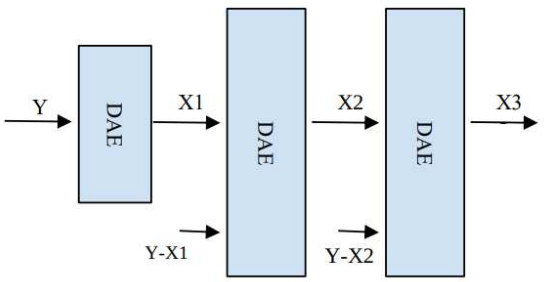


Figure 5.3: Stacked DAE composed of several DAEs with residual connections

between noisy x-vectors and the output of the previous DAE can capture the noise information. Let X be a clean x-vector and Y be a noisy x-vector. If we use Y and $X-Y$ in the input of the denoising autoencoder and construct $\hat{X}$ (the denoised x-vectors) in the output, the results will be very close to X. Moreover, this is similar to residual connection [46] which helps to build deeper models.

5.3. ADDITIVE NOISE COMPENSATION

The first part of our experiments is devoted to additive noise compensation. The x-vector extractor, the noise compensation training data, and the test protocols are described in the following sections.

5.3.1. EXPERIMENTS SETUP

TRAIN X-VECTOR EXTRACTOR

In the experiments, we used TDNN speaker embedding network introduced in Table 5.1. Firstly, we augmented Voxceleb1 with different branches of the Musan corpus (music, babble, noise, reverberation). Then, we extracted MFCC features for 500,000 randomly chosen utterances from the augmented data. Then, Cepstral Mean Variance Normalization (CMVN) is applied to the features. Finally, the VAD is applied to remove silent frames. The trained network is used in all experiments to create x-vectors in train and test parts of denoising techniques.

TRAIN AND TEST X-VECTORS USED IN DENOISING TECHNIQUES

In denoising techniques, we need the noisy x-vectors and their corresponding clean version. Simulating noisy signals by adding noise to clean signals in the time domain is an approach that has proven its effectiveness for generating training data in an x-vector extractor (data augmentation). However, in this chapter, we use the same approach to generate training and test data used in denoising techniques.

For training x-vectors, we used Voxceleb1 and Voxceleb2. Firstly, we extracted the x-vectors from clean files in Voxceleb1 and Voxceleb2. Then, BBC Noises and Musan were added to Voxceleb1 and Voxceleb2 with different SNRs from 0 to 15 and the x-vectors of noisy files were extracted. We used 1638 noise files from BBC corpus. The final version of the train data consists of 1.975 million pairs of noisy x-vectors and their corresponding clean version. For some clean files, there is more than one corrupted version.

For the test and enrollment dataset, we used the Fabiole corpus. We used 6882 utterances from Fabiole where half of them were used for enrollment and the remaining part used as the test dataset. Because in real applications it is possible to have clean data for enrollment, in our experiments we used clean files for enrollment. But for the test part, we added 547 different noises from the BBC corpus to the test files with different SNRs from 0 to 15. The noise files used in the test are different from those used in the training dataset. Since the length of utterances in Fabiole

is varied from very short (less than 2 seconds) to longer utterances (more than 12 seconds), we separated the utterances into 6 groups to investigate the results of denoising methods on each group and especially to observe the effectiveness of the denoising techniques on very short utterances.

SCORING METHOD

In the experiments, the PLDA classifier is used. Before training the PLDA, the x-vectors are centered, and their dimensionality is reduced to 128 with LDA. The PLDA is trained with 200k utterances chosen randomly from Voxceleb. Both clean files and their augmented versions are used.

5.3.2. RESULTS

In this section, we describe the results obtained by the statistical i-MAP method and our proposed methods. The results are presented in Table 5.2 and Table 5.3. In our experiments, we used the equal error rate (EER) metric to evaluate the performance of the recognition system.

Clean: In this experiment, the clean version of x-vectors is used in the test dataset. As it is shown, the results are strongly dependent on the duration of test files. The aim of this experiment is to compare the results obtained by denoising techniques with noise-free x-vectors.

Noisy: To show the weakness of x-vectors in noisy environments, the BBC noise files were added to the test data. In Table 5.2, we can see that there is a drastic degradation in our results. For example, for utterances longer than 12 seconds the EER increased from 0.83 to 5.13.

i-MAP: In this experiment, we tried to find a denoised version of test x-vectors by using Equation 5.5. As it is shown in Table 5.2, in all cases the i-MAP improves the results significantly. For utterances longer than 12 seconds, it gives 50% improvements in terms of EER. It can also be observed that the best gains are obtained for longer segments.

DAE: The network is trained with a stochastic gradient descent algorithm with a learning rate equal to 0.02 and 0.0001 decay of the learning rate. The number of epochs in this experiment and all other variations of DAEs set to 100, and Mean Square Error (MSE) is used as a loss function. From our experiments, we observed that the number of epochs is very important and a small reduction in MSE value has a large impact on the results. In Table 5.2, we can see that for shorter utterances the fine-tuned DAE outperforms the i-MAP method but for longer utterances, the results

Table 5.2: Additive noise compensation by i-MAP, DAE, DAE i-MAP, and Gaussian DAE (EER)

Duration	[0,2]	[2,4]	[4,6]	[6,8]	[8,10]	[10,12]	[12,]
Clean	11.59	7.64	4.14	2.23	3.11	1.53	0.83
Noisy	15.94	12.88	10.50	7.83	8.88	6.66	5.13
i-MAP	14.2	11.07	8.011	5.59	5.33	4.10	2.63
DAE	13.62	10.87	8.28	5.59	5.77	4.10	2.69
DAE i-MAP	14.78	10.87	8.28	4.85	5.33	3.59	2.75
Gaussian DAE	14.20	9.85	7.45	5.59	5.77	4.103	3.14

Table 5.3: Additive noise compensation by Deep Stacked DAE (EER)

Length/ Nř DAEs	[0,2]	[2,4]	[4,6]	[6,8]	[8,10]	[10,12]	[12,]
Clean	11.59	7.64	4.14	2.23	3.11	1.53	0.833
Noisy	15.94	12.88	10.50	7.83	8.88	6.66	5.13
2	13.04	10.46	8.011	5.22	5.33	3.59	2.50
3	13.04	9.65	7.73	4.85	5.77	4.10	2.50

are equivalent to i-MAP ones.

DAE i-MAP: In this method, we used the DAE with the same architecture and parameters used in simple DAE. The denoised x-vectors are passed through the i-MAP method for further noise compensation. In several cases, the results with this hybrid system are better than simple DAE. In Table 5.2. we can see that for utterances between 10 and 12 seconds, we achieved 46% improvement in relative EER.

Gaussian DAE: In this experiment, we try to not only decrease the MSE with DAE but also maximize the prior probabilities of the estimated noise and of the obtained x-vectors. The only difference between Gaussian DAE and simple DAE is the loss function. In this experiment, the function defined in equation 5.2.5 were used as loss functions. The results in Table 5.2 show that in some cases, the Gaussian DAE gives better results than i-MAP and conventional DAE.

Deep Stacked DAE: In this experiment, the Deep Stacked DAE which was introduced in section 5.2.6 is used. In the first DAE, there are three layers. The input and output layers are linear and the hidden layers activation function is $tanh$. The output of the first DAE concatenated with the difference between the noisy x-vector and the output layer from its predecessor DAE, fed to the next DAE. In the second DAE, there are two $tanh$ layers with 1024 neurons and the output layer is linear. The number of neurons in the output layer is 512 which equals the length of a noisy x-vector. The network

was trained with the stochastic gradient descent method. The learning rate is 0.02 and the decay of the learning rate is 0.0001. In another experiment, three stacked DAEs are used. The architecture of the third DAE is the same as the second one. The third DAE receives its input from the output of the second DAE concatenated with the difference between noisy x-vectors and the output layer of the second DAE. The results in Table 5.3 show that adding more DAEs doesn't bring significant improvement.

As we can see from Table 5.2 and Table 5.3, there is a slip in the results for utterances between 8 and 10 seconds. We believe that it happens because the number of trials for this experiment in the Fabiole corpus is small.

5.4. DATA AUGMENTATION VERSUS NOISE COMPENSATION

A number of studies [58–60] emphasized the importance of increasing the number of speakers and data augmentation. It is shown that increasing the number of speakers and using data augmentation makes the x-vector system more robust on noisy and far-field test data. We will show that while data argumentation and increasing the number of speakers make the x-vector system more robust, we can go further and achieve better results with noise compensation techniques. We show that even with large augmented data and a great number of speakers, noise compensation techniques are effective.

To do that, we train two x-vector systems. In the first one, the x-vector network is trained with Voxceleb1, and in the second one the network is trained with a combination of Voxceleb1 and Voxceleb2. In both cases, the train data is augmented with all branches of the Musan corpus. We show that in both protocols the relative gain of EER after denoising x-vectors is significant. Hence, using denoising techniques even with the availability of huge data is a good solution to increase the robustness of speaker recognition systems.

We use two different denoising techniques. Firstly, the denoising autoencoder proposed in section 5.2.4 is applied for noise compensation. Then we apply the Deep Stacked DAE that is introduced in section 5.2.6.

5.4.1. EXPERIMENTS SETUP

B. TRAIN X-VECTOR EXTRACTOR

In our experiments, we used the standard Kaldi[2] x-vector extractor introduced in [2]. The architecture of the TDNN network is shown in Table 5.1. The training data

[2]https://kaldi-asr.org/

is augmented with different branches of the Musan corpus (music, babble, noise, reverberation). Then, we extracted MFCC features for the augmented data. The MFCC features are normalized by Cepstral Mean Variance Normalization (CMVN) and silent frames are removed by the VAD. To explore the reliance of the x-vector system on the number of speakers and utterances in noisy environments, we trained two different networks. The first one is trained with 500,000 augmented utterances from Voxceleb1. In the second one, 1,000,000 randomly chosen utterances from Voxceleb1 and Voxceleb2 were used. In each protocol, the trained network is used to extract train, and test x-vectors that are used in denoising techniques.

C. TRAIN AND TEST X-VECTORS USED FOR DENOISING TECHNIQUES

In denoising techniques, we need the pairs of noisy-clean x-vectors. We use two protocols to see the effectiveness of denoising techniques on x-vectors extracted from a network trained with poor data (Voxceleb1) and a network trained with more rich data (Voxceleb1 + Voxceleb2). The details about training and test dataset used in denoising techniques are described in the following.

Protocol1: In this protocol the x-vectors are extracted by the network trained with Voxceleb1. Firstly, the x-vectors for clean files in Voxceleb1 and Voxceleb2 are extracted. The BBC Noises and Musan were added to Voxceleb1 and Voxceleb2 with different SNRs from 0 to 15 to create the corresponding noisy x-vectors for each clean file. We used 1638 noise files from BBC corpus. The train data consists of 1.975 million pairs of noisy-clean x-vectors. It deserves to be mentioned that for some clean files, there is more than one noisy version. For the test and enrollment dataset, the Fabiole Corpus is used. The Fabiole corpus includes 6882 utterances that 3441 files were used for enrollment and the remaining part was used as the test dataset. The test utterances were corrupted with 547 different noises from the BBC corpus with different SNRs between 0 and 15. Since the length of utterances in Fabiole is varied from very short (less than 2 seconds) to longer utterances (more than 12 seconds), we separated utterances by their duration in 6 groups to see the results of denoising methods on each group and especially to observe the effectiveness of the denoising techniques on very short utterances.

Protocol2: In this protocol the x-vector network is trained with Voxceleb1, Voxceleb2, and one million utterances from Voxceleb1, and Voxceleb2 augmented with different parts of Musan corpus (Noise, Babble, Speech) and real RIRs. The noise compensation training dataset includes 1,200,000 pairs of noisy-clean vectors from Voxceleb1 and Voxceleb2. The added noises are the same as protocol1. In this dataset for each noisy x-vector, there is only one clean version. The test and enrollment files extracted by

this network are the same as protocol 1.

5.4.2. RESULTS

In this section, we describe the results of experiments for the baseline systems and denoising techniques. The results are summarized in Table 5.4 and Table 5.5. In the experiments, the equal error rate (EER) metric is used to evaluate the performance of the speaker recognition system. In all experiments, the PLDA classifier is used for scoring.

Clean: In this experiment the scoring is done on clean x-vectors in the test dataset. We can see that the results are strongly dependent on the duration of test files.

Noisy: To see the performance of the x-vector system in noisy environments, the BBC noise files were added to the test data. In Table 5.4, we can see that there is a drastic degradation in our results. For example, in Protocol1 for utterances longer than 12 seconds the EER increased from 0.833 to 5.131. From Table 5.5 we can see that increasing the number of speakers and a number of training data makes the system more robust but still, there is a large drop in the performance of the system after adding the noise to the test data set. For example, in utterances longer than 12 seconds the EER increased from 0.5% to 2.69%.

Denoising autoencoder: Finding a good architecture and its parameters for a specific problem is the main challenge of using denoising autoencoders. In our experiments, we used a denoising autoencoder with three layers. The input and output layers activation functions are linear. The hidden layer has 1024 neurons with a tanh activation function. The network is optimized by a stochastic gradient descent algorithm. The learning rate was 0.02 which decays 0.0001 at each epoch. The network is trained in 100 epochs to reduce the mean square error (MSE) loss function. In all experiments with conventional DAE and its modifications in the next experiments, we used Tensorflow [130] and Keras [131] frameworks. From Table 5.4 and Table 5.5 we can see that in all cases the denoising autoencoder improves the performance of the system in terms of EER. In Protocol1 we have 14% to 47% relative improvement of EER. This improvement in Protocol2 is from 19% for utterances less than 2 seconds to 58% for utterances between 8 and 10 seconds. The improvement for utterances longer than 10 seconds is 52%.

Deep Stacked DAE: In this experiment, we used Deep Stacked DAE. We used two DAE blocks. In the first one, we put three layers. The input and output layer is linear and a dense layer with 1024 neurons was used in the hidden layer with $tanh$ activation function. The output of the first DAE block concatenated with the difference between noisy x-vector and the output layer from the first DAE. This concatenated

vector is used in the input of the next DAE block. In the second DAE, we used two $tanh$ layers with 1024 neurons and the output layer is linear. The number of neurons in the output layer is 512 which is equal to the size of the noisy vector. The stochastic gradient descent optimization method is used to train the network. The learning rate is 0.02 and the decay of the learning rate is 0.0001.

From Table 5.4 and Table 5.5, we can see that in all experiments the Deep Stacked DAE outperforms the simple DAE. In Protocol1, we have an 18% relative improvement for utterances shorter than 2 seconds and 51% improvement for utterances longer than 12 seconds. In Protocol2, we have a 21% improvement for utterances shorter than 2 seconds and a 66% improvement for utterances between 8 and 10 seconds. The results show that even with a smaller number of training samples in denoising techniques the improvements in protocol 2 are higher.

Table 5.4: The results for x-vectors extractor trained with Voxceleb1 (Protocol1) and denoising techniques

Duration	[0,2]	[2,4]	[4,6]	[6,8]	[8,10]	[10,12]	[12,]
Clean	11.59	7.64	4.14	2.23	3.11	1.53	0.83
Noisy	15.94	12.88	10.5	7.83	8.88	6.66	5.13
DAE	13.62	10.87	8.28	5.59	5.77	4.10	2.69
Stacked DAE	13.04	10.46	8.01	5.22	5.33	3.59	2.50

Table 5.5: The results for the x-vector extractor trained with Voxceleb1+Voxceleb2 (Protocol2) and denoising techniques

Duration	[0,2]	[2,4]	[4,6]	[6,8]	[8,10]	[10,12]	[12,]
Clean	10.43	4.62	1.93	1.11	0.88	1.02	0.57
Noisy	13.62	9.65	7.18	5.22	5.33	3.07	2.69
DAE	11.01	7.04	4.42	3.35	2.22	1.538	1.28
Stacked DAE	10.72	6.43	3.86	2.61	1.77	1.53	1.28

From the results in Table 5.4 and Table 5.5, we observed that, by increasing the number of speakers and using more data in training the x-vector network, the system becomes more robust in dealing with unseen noises. But the performance drops significantly in comparison to noise-free environments. Applying noise compensation techniques brings notable improvement in both protocols. The relative improvement of EER for both protocols is presented in Figure 5.4.

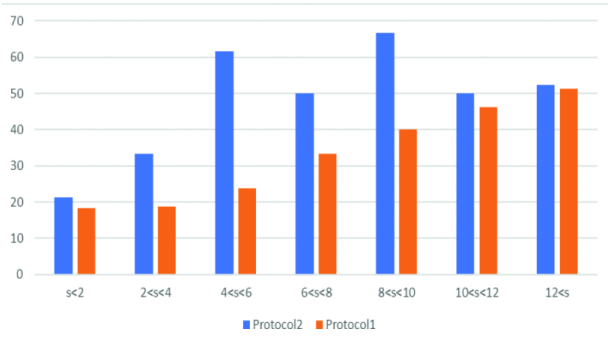


Figure 5.4: The relative improvement of EER(%) by deep-stacked denoising autoencoder in Protocol1 and Protocol2

5.5. SPECIFIC NOISE COMPENSATION

Having a general noise compensation module that works with all unseen noises is more convenient than adapting it to different environments. In this part of our experiments, we explore the adaptation of noise compensation for a specific noise. The intention behind this experiment is to answer whether having a general noise compensation is sufficient to achieve a high-performance speaker recognition system or whether we need to have an adapted noise compensation module for a specific situation. In this experiment, the TDNN x-vector extractor is used (Table 5.1), and the Deep Stacked DAE is used as a noise compensation module (Section 5.2.6).

5.5.1. EXPERIMENTS SETUP

x-vector extractor: The TDNN is trained with Voxceleb1, Voxceleb2, and one million utterances from Voxceleb 1 and Voxceleb 2 augmented with different parts of Musan corpus (Noise, Babble, Speech) and real RIRs.

noise compensation training data: We used two sets of training data. The general model is trained with pairs of noisy/clean x-vectors extracted from Voxceleb1,2. In this case, the random noises from the BBC corpus are added to the clean version. In the second model that is trained to compensate for a specific noise, all files are corrupted with the specific noise. This noise is the same in training and test data.

test and enrollment: The Fabiole corpus is used as test and enrollment. The 3441 files were used for enrollment and the remaining part was used for enrollment. The test files are corrupted with motorcycle noise chosen from the BBC noises.

5.5.2. RESULTS

In Table 5.6 we see the impact of motorcycle noise in different SNRs. As it is shown in low SNR=2 the EER increase from 0.57 to 4.01. This degradation in high SNR is not so significant. We are using the lowest SNR to be compensated with both general and specific noise compensation modules.

Table 5.6: TABLE 2. The impact of specific noise at different SNRs (EER)

SNR/Dur	[0,2]	[2,4]	[4,6]	[6,8]	[8,10]	[10,12]	[12,]
Clean	10.43	4.628	1.934	1.119	0.888	1.026	0.577
2	9.275	6.439	7.182	5.97	6.222	5.128	4.105
5	9.275	5.835	5.249	4.478	4.444	3.077	1.988
10	9.855	5.03	3.315	2.612	1.778	1.538	0.9622
15	9.855	5.03	2.762	1.493	1.333	1.026	0.6414

The results for specific noise compensation are presented in Table 5.7. The third row shows the output of specific noise compensation and the last row is the EER obtained from general noise compensation. In the case of long duration, we will see that targeting a specific noise gives more improvement. However, the obtained results with the general model are competitive with the specific one. These results show that Deep Stacked DAE is learning a complex hyperplane between noisy and clean distributions for a big number of noises. This characteristic helps us to have a general noise compensation module for all unseen noises.

Table 5.7: Specific noise compensation (EER)

Duration	[0,2]	[2,4]	[4,6]	[6,8]	[8,10]	[10,12]	[12,]
Clean	10.43	4.628	1.934	1.119	0.888	1.026	0.577
SNR=2	9.275	6.439	7.182	5.97	6.222	5.128	4.105
Specific	11.3	6.036	3.867	3.358	3.111	1.538	1.475
General	9.855	5.634	3.867	2.985	3.111	2.051	1.604

5.6. COMPENSATE MULTIPLE DISTORTIONS

In the previous section, we explored the effectiveness of compensating additive noise at the x-vector level. In some cases, there is only one kind of domain mismatch like additive noise or reverberation, but in many cases, there is more than one distortion. Finding a solution for domain adaptation in the presence of different distortions is a

challenge.

In this section, we investigate the situation in which there is none, one or more of the following distortions: early reverberation, full reverberation, and additive noise. We propose two configurations to compensate for these distortions. In the first one, a specific denoising autoencoder is used for each distortion. In the second configuration, a denoising autoencoder is used to compensate for all of these distortions simultaneously.

Indeed, the proposed DAE tries to learn the relation between the x-vector affected by a given distortion and its clean version. Thus, it is direct learning of the distortion's effect, which is expected to make the denoising system more efficient than multi-condition training. The DAE uses more specific information about the distortion than what would be used in a data augmentation approach. Moreover, the DAE we propose is trained in the x-vector space, which reduces the training time, with respect to the x-vector extractor training.

5.6.1. SYSTEM CONFIGURATION FOR MULTIPLE DISTORTIONS

Our study will focus on 5 conditions and their combinations: Clean (D), additive Noise (N), Early reverberation (E), Full reverberation (F), and additive noise with Full reverberation (FN). The reverberation is the sum of sound reflections arrived at a single point inside an acoustical enclosure. Early reflections which are called early reverberation arrive between 50-100ms after the arrival of the direct signal. The late reverberation is the next echo that arrives to the listener with longer delays.

In the next sections, the DAE(N), DAE (E), DAE (F), DAE (FN), and DAE (N+E+F+FN) stands for experiments that the input of denoising autoencoder is noisy x-vector data, early reverberated x-vector data, the full reverberated x-vector data, the simultaneously noisy fully reverberated x-vector data and finally a combination of x-vectors for all distortions. The output of the DAE is always the clean x-vector data.

In this part, we compare two approaches for handling multiple distortions. In the first approach, after the x-vector network, a specific denoising autoencoder is used for each distortion. For clean x-vectors, the scoring is done without passing them through a DAE. The details of this configuration are depicted in Figure 5.5. In this configuration, we assume that the type of distortion is known. We will show when the type of distortion is unknown, using a classifier can help to detect the kind of distortion automatically.

In the second configuration depicted in Figure 5.6, the compensation for different distortions is done by using one DAE. In this configuration like the specific

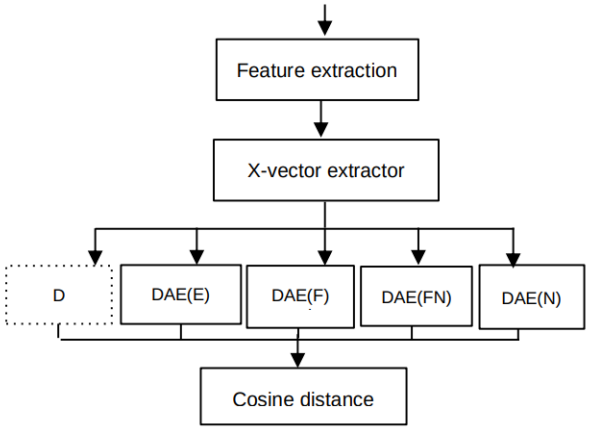


Figure 5.5: Using specific models for each distortion

compensation, there is clean speech, distorted speech with additive noise, early reverberation, full reverberation, additive noise, and full reverberation. With this system, it is not necessary to have previous information about the kind of distortion. As it is shown in the results, the denoising autoencoder learns to compensate for all those distortions simultaneously. In the case of clean speech, we show that, without any change in the system and without having previous information about the environment, the denoising part does not have a negative effect on clean x-vectors. In both configurations, the Stacked DAE is used for noise compensation. The Stacked DAE is implemented in Pytorch[3] library.

5.6.2. NOISE AND REVERBERATION DATA SIMULATION

To train the DAE for dereverberation or for denoising we need to have a set of x-vector pairs, distorted/clean. Distortion here corresponds to reverberation or additive noise. The reverberated version of the speech clips is obtained by convolving the original speech clips with room impulse responses (RIR) simulated with the pyroomacoustics [4]. The RIR was designed to simulate rooms whose dimensions are sampled randomly between $[3m * 4m * 2.5m]$ and $[6m * 8m * 3.5m]$. The reverberation time for the rooms (RT_60) is drawn randomly between $[200ms]$ and $[600ms]$. The microphone and the speech source are located at least $1m$ away from any wall. The microphone is at 0.5m height (to simulate a small robot on the floor) and the speech source height is drawn randomly in $[1.6m, 1.9m]$ (to simulate a human standing). The distance

[3] https://pytorch.org/
[4] https://github.com/LCAV/pyroomacoustics

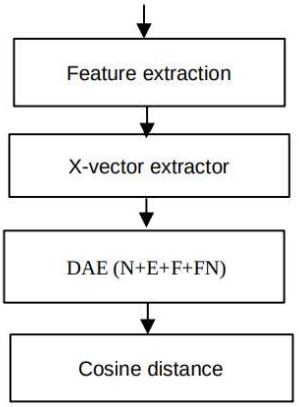


Figure 5.6: General domain adaptation for different distortions

between the speaker and the microphone is at least 1 meter. We generated 10000 RIR for training and 2000 RIR for the test. When considering only early reverberation the RIR is truncated to 50ms after the RIRs first peak. When additive noise is present, the noise source clips are office noises collected from Freesound [122]. We collected 3275 clips for training and 1000 clips for the test. The training/test split is designed such that there is no overlap in terms of Freesound users between both sets. The original noise clips are drawn randomly and convoluted with an RIR file. The noise source is located at least $1m$ away from any wall at a height of $[1.6m, 1.9m]$. The noise files are added with random SNR between $[0, 10]$.

5.6.3. EXPERIMENTAL SETUP

The x-vector network introduced in Table 5.1 was used to create x-vectors for training and testing data. The x-vector network has been trained with Voxceleb1, Voxceleb2, and one million utterances from Voxceleb 1 and Voxceleb 2 augmented with different parts of the Musan corpus (Noise, Babble, Speech) and real RIRs. The x-vector extractor has been used to create clean, noisy, and reverberant data to train DAE.

For each experiment firstly the reverberation and/or noise were added to Voxceleb2. Then pairs of distorted/clean x-vector were produced to train DAE. In the same manner, the data was distorted for the test. In all cases the enrollment data is clean; because we assume that in real applications it is possible to have clean data for enrollment. In the experiments, we use Deep Stacked DAE. The architecture (number of layers and neurons) is the same in all experiments. In the back end, the cosine distance is used for scoring.

The Fabiole corpus was used for test and enrollment. In the Fabiole corpus, there are 6992 files from 130 speakers. From 130 speakers in Fabiole, for 100 speakers there is a small number of utterances. We used these speakers just in enrollment. The utterances belonging to the remaining 30 speakers are randomly separated for the test and enrollment. In enrollment, there are 3576 files and 3244 files are used for the test. Because the duration of files in the Fabiole corpus spans from very short to long, the test files are separated into seven groups every two seconds.

5.6.4. RESULTS

The results obtained in the presence of each distortion are summarized in Table 5.8. When we are in a clean environment without noise or reverberation, we show that the use of DAE(F+N) gives almost the same (sometimes better) results as the baseline system. This is an interesting property of the proposed approach. If the x-vector belongs to the clean class, the scoring could be done directly. But for the general configuration presented in Fig 5.6, we don't care about the cleanness of the environment and even in the case of noise-free and non-reverberant environments the test x-vector will be passed through the DAE. When we apply the DAE trained on noisy x-vectors on clean x-vectors, the output is still almost identical to the input. It means that for noise and reverberation-free environments, the system could be used without any modification

In the case of additive noise, both specific models and general models were tested. As is shown the results obtained by specific models are a little better. For example, for utterances longer than 12 seconds, the EER obtained by the specific model is 1.91 but the EER obtained by the general model is 2.23. For early reverberation and full reverberation distortions, the results obtained by the general model for short segments are better than the specific model but the results obtained for longer utterances are almost the same. When there is additive noise and reverberation, in all cases the results achieved by the general model are better. For specific models, the experiments are done directly without using a classifier. But to prove that it is possible to detect the type of distortion, we trained a feed-forward neural network. The accuracy of the trained network is 81%. Even if we had a distortion classifier with 100% accuracy, the results show that it's better to use a general DAE instead of using a specific DAE for each distortion

Our experiments show that, in the co-existence of noise and reverberation, the second configuration gives better results. For example, with the second configuration, we obtained 76.6% relative improvement of EER for utterances longer than 12 seconds. For other situations in the presence of only one distortion, the second configuration

gives almost the same results achieved by using a specific model for each distortion.

To explore the reasons behind the effectiveness of the general model, DAE (N+E+F+FN), we did another set of experiments. In these experiments, we explored if it is possible to do compensation for distortions interchangeably or not. When the trained DAE with reverberated data, DAE(F), was used for additive noise compensation, the EER for utterances longer than 12 seconds, was reduced from 6.63 to 5.42. Also, if we use DAE (N) for full dereverberation, the EER for utterances longer than 12 seconds, reduces from 5.23 to 4.29. The results show that using training data from other distortions has a positive impact on training the general model.

Table 5.8: Compensating multiple distortions: D(Dry signal), N(Additive noise), E(Early Reverberation), F(Full reverberation), FN(Additive noise and Reverberation) (EER)

Env	Duration	[0,2]	[2,4]	[4,6]	[6,8]	[8,10]	[10,12]	[12,]
D	D	9.89	5.08	3.62	2.44	1.77	1.85	1.65
	DAE (FN)	9.01	5.06	3.31	2.40	1.75	1.85	1.65
N	N	14.58	11.16	8.83	9.12	8.05	7.97	6.63
	DAE (N)	10.48	6.08	3.56	3.27	2.26	1.91	1.91
	DAE (N+E+F+FN)	10.75	6.93	4.45	4.47	3.53	2.77	2.23
E	E	18.60	10.54	7.08	4.58	3.98	4.16	4.02
	DAE (E)	13.08	6.48	4.15	2.50	2.23	2.31	2.29
	DAE (N+E+F+FN)	11.57	5.67	3.85	2.96	2.58	2.77	2.29
F	F	20.01	11.96	8.01	6.15	4.88	5.09	5.23
	DAE (F)	9.05	5.03	3.26	2.48	2.21	2.31	2.23
	DAE (N+E+F+FN)	9.29	4.86	3.62	2.49	2.21	2.31	2.05
FN	FN	27.32	24.34	20.77	18.68	19.38	19.93	17.61
	DAE (FN)	14.24	9.93	7.37	4.97	3.96	3.68	4.53
	DAE (N+E+F+FN)	13.66	9.73	6.52	4.54	3.61	3.24	4.09

5.7. CONCLUSION

In this chapter, the noise compensation in the TDNN x-vector system is explored extensively for several scenarios. Firstly, we showed that in x-vector space when there are many unseen noises, the results degrade substantially. Then, we tried to exploit i-MAP statistical denoising technique, originally designed for i-vector space, to denoise x-vectors. The i-MAP technique is applicable in the x-vector domain. In this chapter, several variations of denoising autoencoders are proposed. The combination

of i-MAP and DAE is better than using them separately for denoising x-vectors. In another method (Gaussian DAE), we defined a new loss function that tries to combine the MSE term with two additional terms corresponding to the likelihood of the output data with respect to prior Gaussian distributions. Finally, a Deep Stacked DAE was proposed that each DAE receives the output of its predecessor DAE concatenated with the difference between noisy x-vectors and its predecessors output. The results obtained by Deep Stacked DAE, outperform the statistical i-MAP technique and other variations of DAE discussed in this chapter.

We showed that the system's performance continues to improve by increasing the number of speakers and data argumentation. We showed also that even with this fact, applying compensation techniques is essential to approaching noise-free test conditions. The obtained results for specific noise compensation show that having a general Deep Stacked DAE is competitive with specific noise models.

Finally, we proposed two configurations for robust speaker recognition in environments where there are several distortions. The systems should perform efficiently in environments with early reverberation, full reverberation, additive noise, additive noise, and full reverberation. To solve this problem, we proposed two configurations. In the first configuration, we used a specific DAE for each distortion. In the second configuration, one DAE is used to learn all of these distortions simultaneously. The second configuration is simpler and gives the same or even better results than specific compensation for each distortion. We also showed that the speaker recognition performance doesn't decrease (with respect to the baseline) when the test data is clean, which is a positive behavior of the proposed noise compensation module.

6

EXPLORING THE BEHAVIOR OF RESNET SPEAKER RECOGNITION SYSTEM AGAINST ADDITIVE NOISE AND REVERBERATION

In the previous chapter, the behavior of TDNN speaker recognition systems against additive noise and reverberation has been explored. In this chapter, we extend our study to the ResNet speaker recognition system. Firstly the robustness of the ResNet in the presence of noise, reverberation, and both distortions is explored. Our experimental results show that in all cases the ResNet system is more robust than TDNN. After that, a noise compensation task is done with a denoising autoencoder (DAE) over the x-vectors extracted from both systems. We explored two scenarios: 1) compensation of artificial noise with artificial data, and 2) compensation of real noise with artificial data. The second case is the most desired scenario because it makes noise compensation affordable without having real data to train denoising techniques. The experimental results show that in the first scenario noise compensation gives significant improvement with TDNN while this improvement in ResNet is not significant. In the second scenario, we achieved a 15% improvement of EER over the VoiCes Eval challenge in both TDNN and ResNet systems. In most cases, the performance of ResNet without compensation is superior to TDNN with noise compensation.

The presented work in this chapter is published in [54] and [132].

6.1. INTRODUCTION

THERE is bulky research on proposing different speaker embedding networks to improve the quality of speaker representations in general. This line of research aims to improve the quality of speaker recognition systems for all environments. In Chapter 2 the main proposed DNNs including TDNN [2], ResNet [46], ECAPA-TDNN [5] and MFA-Conformer [4] reviewed. The main task of these networks is the same: extract a fixed-size compact representation from variable-length speech utterances known as speaker embedding or x-vector.

We observe an evolution in the proposed architectures. The ResNet system leverages a high number of hidden layers. The residual connections are used to make its training feasible. This big number of hidden layers with the ability to capture context information by conventional layer and efficient objective function has improved the performance of this system more significantly in comparison to the TDNN system. The effectiveness of the ResNet system is not limited to noise-free environments but it is more promising against environment variabilities such as additive noise, reverberation, and far-distance recording devices [120] [3].

The previous research shows the weakness of TDNN-based speaker recognition systems against noise and reverberation distortions. In Section 5.3 and Section 5.6 in Chapter 5 it is shown that in the presence of noise and reverberation using a compensation module before scoring (statistical or DAE) can bring the performance of the x-vector system closer to a clean situation. Several strategies such as data argumentation (Section 5.4) and noise compensation are explored to make the TDNN-based speaker recognition systems more robust against noise, reverberation, and other variabilities (Section 5.3 and Section 5.6).

In this chapter, we first explore the robustness of the ResNet speaker embedding system in different situations. After that, we explore the possibility of doing compensation for environment variabilities with the ResNet speaker embeddings. Throughout our work, we compare the performance of ResNet with the TDNN system which is the most used and known speaker embedding system [2]. In our research, we will show that even in difficult situations in the presence of additive noise with low SNR and reverberation, the ResNet system is more robust than the TDNN system, but still, we face some performance degradation in noisy and reverberated environments.

Despite the efficiency of noise compensation in TDNN, we found no or small improvement in EER by using different statistical and DNN compensation techniques in the ResNet system. Here, a serious question arises in regard to the behavior of ResNet against noise and reverberation. It is not clear whether the noise and reverberation are compensated internally by the ResNet system or if there is another

characteristic that impedes us from doing compensation. In an attempt to understand the reasons behind the behavior change of noise compensation in ResNet, the t-SNE visualization of clean x-vectors and their noisy versions is studied for both systems.

Our visualization shows that the ResNet speaker embedding extractor performs the compensation to a great extent and even in the presence of noise and reverberation the extracted speaker embeddings remain close to the clean environment. This is in conformity with ResNet results obtained from the severely noisy and reverberated environments in comparison to the clean environment. However, the possibility of doing compensation for the residual small degradation of performance in the presence of acoustic noises remains an open question in the ResNet system.

6.2. SYSTEM CONFIGURATION

In this section, the architecture of the ResNet speaker embedding network is presented. After that, the integration of the compensation module with the speaker recognition system is presented.

6.2.1. RESNET AND TDNN ARCHITECTURE

The ResNet speaker embedding extractor used in this chapter is a variant based on ResNet [46]. The ResNet model for extracting x-vectors is made of three parts: a *frame-level* feature extractor, a *statistics-level* layer, and *segment-level* representation layers.

- The frame-level component is based on the well-known ResNet topology. ResNet (Residual Network) uses a stack of many Residual Blocks. A Residual Block is made up of two 2-dimensional Convolutional Neural Networks (CNN) layers separated by a non-linearity (ReLU). The input of the Residual Block is added to its output in order to constitute the input of the next Residual Block. Residual blocks are allowed to solve the problem of exploding and vanishing gradients. And ResNet has emerged as a family with extremely deep architectures showing compelling accuracy and effective convergence behaviors.
- The statistics-level component is an essential component that converts from a variable-length speech signal into a single fixed-dimensional vector. The statistics level is composed of one layer: the statistics-pooling, which aggregates over frame-level output vectors of the DNN and computes their mean and standard deviation.
- The segment-level component maps the segment-level vector to speaker

Table 6.1: The proposed ResNet-34 architecture. In the last row, N is the number of speakers. The dimensions are (Frequency×Channels×Time). The input is comprised of 60 filter banks from speech segments. During training, we use a fixed segment length of 400.

Layer name	**Structure**	**Output**
Input	–	60 × 400 × 1
Conv2D-1	3 × 3, Stride 1	60 × 400 × 32
ResNetBlock-1	$\begin{bmatrix} 3\times3,32 \\ 3\times3,32 \end{bmatrix} \times 3$, Stride 1	$60\times400\times32$
ResNetBlock-2	$\begin{bmatrix} 3\times3,64 \\ 3\times3,64 \end{bmatrix} \times 4$, Stride 2	$30\times200\times64$
ResNetBlock-3	$\begin{bmatrix} 3\times3,128 \\ 3\times3,128 \end{bmatrix} \times 6$, Stride 2	$15\times100\times128$
ResNetBlock-4	$\begin{bmatrix} 3\times3,256 \\ 3\times3,256 \end{bmatrix} \times 3$, Stride 2	$8\times50\times256$
Pooling	–	8×256
Flatten	–	2048
Dense1	–	256
Dense2 (Softmax)	–	N
Total	–	–

identities. The mean and standard deviation are concatenated together and forwarded to additional hidden layers and finally to the softmax output layer

The stklia is used for ResNet implementation [1]. In our experiments, we used the TDNN architecture introduced in Section 5.2.1.

6.2.2. COMPENSATION MODULE

The compensation module performs a transformation between noisy and clean speaker embeddings. In doing so, the compensation module tries to remove the impact of noise from speaker embeddings. In this work, two techniques are tested for compensation. We are using the Deep Stacked DAE for noise compensation. The architecture of the Deep Stacked DAE is described in Section 5.2.6. In the experimental phase, we tested several variants of this architecture for noise compensation in the ResNet system. These details will be discussed in the next section.

[1] https://github.com/Chaanks/stklia

6.3. EXPERIMENTAL SETUP

6.3.1. SPEAKER EMBEDDING EXTRACTOR TRAINING

The speaker embedding extractors are trained on the Voxceleb2 corpus. In order to increase the diversity of the acoustic conditions in the training set, the MUSAN corpus was used for data augmentation [121]. Also, an RIR pool is used for data reverberation [2]. The TDNN system is trained with MFCC features with 25 frame lengths, and ResNet is trained on 60 filter banks with 25 frame lengths.

6.3.2. TEST AND ENROLLMENT

In our experiments, we used two datasets. Fabiole protocol is used to evaluate the robustness of the system against simulated noise and reverberation. In the Fabiole protocol, we have 130 hundred speakers in the enrollment and 30 speakers for the test. The number of test files is 6870. In both protocols, one file is used per speaker in the enrollment. The Voices protocol is used to evaluate the robustness against real noise and reverberation. Voices [6] dataset has train and test parts. The test part was created from 1320 clean files coming from Librispeech (100 speakers) and the train part is recorded from 2583 files coming from Librispeech (200 speakers). We used 300 files, each file belonging to one speaker for enrollment, and 3603 remaining files are used as test utterances. In all experiments with Voices, the far microphone (mic 05) and the rooms (room2, room3) with more reverberation are chosen. The details of protocols are presented in Table 6.2.

6.4. BACK-END

The TDNN system is evaluated with PLDA back-end. The PLDA is trained with 200k utterances extracted from Voxceleb. The ResNet is evaluated with cosine similarity.

6.5. RESULTS AND DISCUSSION

6.5.1. EXPLORING THE ROBUSTNESS OF TDNN AND RESNET

In this section, the results are presented. In all experiments, the ResNet is compared with the TDNN x-vector system. The results for different situations are presented in Table 6.3.

Clean. In the baseline experiment, there is no noise or reverberation.

Additive noise. Because there is no dataset with just additive noise, the

Table 6.2: Test protocols derived from Fabiole and Voices evaluation datasets

Protocol	Test	Enroll	Trials
Fabiole	6870	130	893k
Voices	3603	300	1080k

systems are evaluated with simulated additive noise. The experiments are done with Fabiole protocol. The Freesound [122] noises are added to the clean speech with Pyroomacoustics [2] tool. In three experiments, different SNRs are tested.

Reverberation. In another experiment, we explored the robustness of both systems against reverberation. In this case, we tested the systems with both real and simulated reverberation. The protocol for adding reverberation is described in Section 5.6. For real reverberation, we used recorded files of room2 and room3 in Voices that are recorded without noise.

Table 6.3: Robustness od TDNN and ResNet systems against different distortions (EER)

Distortion	Protocol	ResNet	TDNN
Clean	Fabiole	6.27	15.21
	Voices	0.89	1.25
Noise	Fabiole [SNR 0-5]	8.28	17.83
	Fabiole [SNR 5-10]	7.43	16.58
	Fabiole [SNR 10-15]	6.87	15.95
Reverberation	Fabiole	9.75	18.20
	Voices room 2	1.24	2.53
	Voices room 3	2.6	6.68
Reverberation and Noise	Fabiole [SNR 0-5]	12.48	21.47
	Voices room 2	1.24	3.71
	Voices room 3	2.6	6.69

Additive noise and reverberation. In this case, the systems are tested with both real and simulated noise and reverberation.

If we compare the baseline results in Table 6.3 with the presence of distortions, we see that the ResNet has relative robustness in the presence of noise and reverberation. For example, in the worst case for SNR between 0 and 5 and reverberation on Fabiole protocol, the EER is 12.48 while with TDNN in the clean environment, the EER is

[2] https://github.com/LCAV/pyroomacoustics

15.21. With VoiCes protocol in the presence of noise and reverberation in room3, the EER is 2.6 while for the TDNN system, the EER is 6.69. Just in the case of noise and reverberation for Fabiole protocol the performance of ResNet degrades significantly. One possible reason behind this degradation in Fabiole comes from the fact that in Fabiole there are 1720 files shorter than 4 seconds. In all other experiments, ResNet shows relative robustness against noise and reverberation. For example, in the VoiCes protocol in the clean position, the EER is 0.89 but in the presence of severe noise and reverberation, it is 2.6.

6.5.2. NOISE COMPENSATION

In the second group of experiments, we did noise compensation in both TDNN and ResNet systems in the presence of artificial and/or real noises and reverberation. During noise compensation, two scenarios are considered. The results for each scenario are described in this subsection.

ARTIFICIAL NOISE COMPENSATION WITH ARTIFICIAL TRAINING DATA

In our experiments, we used pairs of noisy/clean speaker embeddings to train DAE. The training pairs are constructed from Voxceleb. The noisy version is prepared by adding Freesound noises and RIR files with Pyroomacoustics. In the training data, there are about 5 million pairs of noisy/clean x-vectors. In the noisy version, there are one or both additive noise and reverberation distortions. The noises and RIR files used to prepare the training data are different from those that are used for test protocols.

After doing a transformation on noisy test files with the trained DAE, we observed a small gain in terms of EER for the ResNet system. For example, in the presence of additive noise and reverberation in Fabiole protocol, the EER reduces from 12.48 to 12.18, while in TDNN it reduces from 21.74 to 18.03. The results are shown in Table 6.4.

Table 6.4: Simulated noise compensation in TDNN and ResNet systems (EER)

System	Clean	Noisy	Denoised
TDNN	15.21	21.47	18.03
ResNet	6.27	12.48	12.18

We explain this phenomenon by visualization of noisy and clean speaker embedding with t-SNE. The visualization shows that ResNet x-vectors remain in the same space

and the noise and reverberation don't have a big impact on them. In this experiment, we chose a random noisy speaker embedding and its 1000 closest neighbors. The chosen vectors are plotted alongside their corresponding clean versions. The t-SNE is trained with both clean and noisy speaker embeddings. This experiment shows that noisy speaker embeddings in the ResNet system are not separable and are far from their clean version. But in the TDNN system, there is a significant shift between noisy and clean speaker embeddings. This phenomenon explains that there is no big difference between noisy and clean versions of speaker embeddings to be compensated by denoisers. This is in conformity with obtained EER in a noisy and reverberant environment leveraging the ResNet system (Fig. 6.1). However, the small residual noise in ResNet is not trainable with DAEs, we don't know whether we have arrived at the limit of doing noise compensation in this system or if it is possible to do noise compensation in ResNet speaker embeddings (Fig. 6.1).

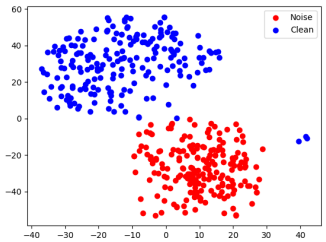
Denoising techniques reduce the MSE between noisy and clean speaker embeddings. We observed that in ResNet we have a small relative gain in terms of MSE between noisy and denoised speaker embeddings, while in the TDNN network, the MSE improves significantly. For example, in the case of noise and reverberation in TDNN, the MSE reduces from 0.21 to 0.07 on the Fabiole test set, while in ResNet this value decreases from 4.18 to 3.94 with DAE. This small relative gain of MSE leads to just a small improvement of EER.

We did several attempts to resolve the problem of noise compensation in ResNet speaker embeddings. In a controlled condition, we tried specific noise compensation by training the noise compensation module using a pair of distorted/clean data that the distorted version is prepared by adding a specific noise. Even in this case, we didn't observe an improvement.

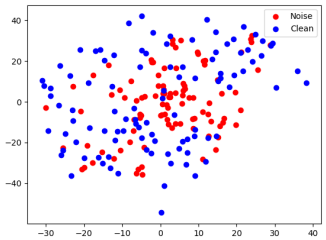
This characteristic of the ResNet shows that the residual noise in this system is more complicated and unpredictable to be modeled by current denoising techniques. However, the small residual noise in ResNet is not trainable with DAEs, we don't know whether we have arrived at the limit of doing noise compensation in this system or if it is possible to achieve better results by noise compensation.

REAL NOISE COMPENSATION WITH ARTIFICIAL TRAINING DATA

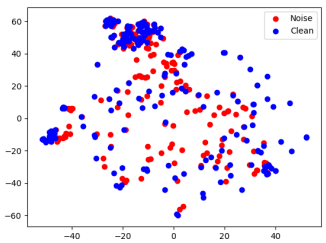
In order to train the DAE for real noise and reverberation compensation, the same training data was used that was prepared already for artificial noise compensation. In this experiment, the standard VoiCes protocols introduced in [6] are used. The results before and after noise compensation are shown in Table 6.5. Intuitively, having simulated training data that matches better with the real noisy test data,



(a) Pairs of clean and noisy embeddings extracted from TDNN system



(b) Pairs of clean and their denoised version embeddings extracted from TDNN system



(c) airs of clean and their denoised version embeddings extracted from ResNet system

Figure 6.1: t-SNE visualization of TDNN and ResNet x-vectors

gives better results. During data simulation, we tried to prepare another training data by fine-tuning several parameters such as room size, sound absorption, and microphone distance. We observed that creating a training simulated data by fixing these parameters doesn't bring more improvement. The experiments show that having more diversified data with different parameters such as random microphone distance and having different room sizes increases the chance of capturing the given noise in the test situation.

Table 6.5: Real noise compensation with artificial noisy training data in TDNN and ResNet systems(EER)

	Voices Eval		**Voices Dev**	
System	**Noisy**	**Denoised**	**Noisy**	**Denoised**
TDNN	4.44	3.80	7.89	7.28
ResNet	1.37	1.15	5.10	5.04

6.6. CONCLUSION

In this chapter, we explored the noise robustness of two state-of-the-art speaker recognition systems: TDNN and ResNet. We have shown through our experiments that the system based on ResNet is much more robust to noise (additive noise and reverberation) than the TDNN. Also, real and artificial noise compensation is done in both systems. The most unexpected result is that the compensation techniques (based on DAE) give a marginal improvement in the case of artificial noises with ResNet, while the improvement is significant in the TDNN system. Despite this finding, the ResNet system remains more efficient than the TDNN, with or without noise, with or without compensation. However we found a degree of improvement in the case of real noise compensation in both TDNN and ResNet systems, we had shown that a precise simulation of real situations is the main challenge of doing real noise compensation with artificial training data. The objective of future work is to handle noise and reverberation in the speaker embedding level in order to avoid the limitations of noise compensation in the speaker embedding level.

7

LEARNING NOISE-ROBUST RESNET SPEAKER RECOGNITION SYSTEM

In the previous chapter, we explored the possibility and limitations of noise compensation at the speaker embedding level for both TDNN and ResNet systems. In the current chapter, we move towards learning noise-robust speaker embedding extractors. This chapter proposes two new variants of ResNet-based speaker recognition systems that make the speaker embeddings more robust against additive noise and reverberation. The goal of the proposed systems is to extract speaker embeddings in noisy environments that are close to their corresponding version in a clean environment. To do so, the speaker embedding network minimizes the speaker classification loss function and the distance between pairs of noisy and clean speaker embeddings jointly. The first proposed system learns the same distribution for both noisy and clean environments. The second proposed system imposes on the speaker embeddings for noisy environment shift towards the distribution of the best-obtained system in the clean environment. The experimental results obtained by our systems are compared with the baseline ResNet system. In different situations with real and simulated noises and reverberation conditions, the modified systems outperform the baseline ResNet system. The proposed systems are tested with four evaluation protocols. In the presence of artificial noise and reverberation, we achieved a 19% improvement in EER. The main advantage of the proposed systems is their efficiency against real noise and reverberation. In the presence of real noise and reverberation, we achieved a 15% improvement in EER.

The presented work in this chapter is published in [133]

7.1. INTRODUCTION

According to the reviewed literature in Chapter 3, the problem of noise and reverberation could be addressed at different levels of speaker recognition systems, including signal level, feature level, speaker modeling level, speaker embedding level, and scoring technique adaptation. Data augmentation is another approach for making speaker recognition systems robust against noise. Researches show that data augmentation brings a degree of robustness to the speaker recognition system, but still, their performance degrades in noisy environments because there is no constraint on the speaker embedding system to extract identical or close speaker embeddings for pairs of the noisy-clean version of the same signal.

In several works, the researchers tried to make the speaker embedding extractors robust to noise and reverberation. In [97] a domain adaptation technique is proposed that uses mean discrepancy distance (MMD) as a regularizer with the speaker embedding extractor that does the adaptation between the source and target domain. The proposed method is tested for language adaptation, and its efficiency for noise and reverberation adaptation is not examined. In [88] an adversarial strategy was proposed to make the speaker embedding extractor more robust against noise. In the general configuration of the speaker embedding extractor, the output layer does speaker classification. In this work, a second classifier is trained adversarially that accepts the type of noise in the output. In another work, a GAN-based speaker embedding extractor was proposed that used a binary discriminator to discriminate the noisiness of the speaker embedding alongside the speaker recognition classifier [87]. The main deficiency of adversarial speaker embedding extractor systems is that the training of the network in a manner that can not be able to discriminate the type of noise or the noisiness of a speaker embedding doesn't guarantee that noisy speaker embeddings are close enough to their clean version. In another words, however, we can have indistinguishable representations for both noisy and clean representations, there is no constraint to guide the training process towards the optimal achievable distribution.

Noise compensation at the speaker embedding level (i.e. extracted speaker embeddings, x-vectors), the estimation of a clean speaker embedding from its corresponding noisy version, by doing a transformation between pairs of noisy/clean speaker embeddings is another approach that performed well in the compensation of artificial noise and reverberation. In Chapter 5 we showed that this approach performs well in some cases, but it doesn't bring a significant improvement with all speaker embedding systems and in all environments. The behavior of different speaker embedding systems is different because they just consider the speaker classification accuracy (inter-speaker and intra-speaker distance) during optimization

and they don't put an explicit constraint on the noise impact. This characteristic makes noise compensation more difficult in some speaker embedding systems such as ResNet. To overcome this challenge, in this chapter, we propose two training strategies of ResNet-based speaker recognition systems that impose on the ResNet to extract speaker embeddings for noisy signals that are close to their corresponding version in a clean environment.

In the first approach, the objective is to optimize the speaker embedding in a manner that converges toward the same point for the pairs of noisy/clean samples. In this system, two loss functions are used. The speaker classifier loss function is minimized and at the embedding layer, the mean square error (MSE) between noisy and clean speaker embeddings is optimized. Although the system improves for noise and reverberation, its performance is lower than the baseline system for clean environments.

To solve this problem, we propose a second system. In this system firstly an optimal speaker embedding extractor for a clean environment is trained. After that, another speaker embedding is trained that jointly reduces the speaker classifier loss function, and mean square error (MSE) between the output of the embedding layer and an optimal clean speaker embedding extracted with the pre-trained system. Since it is imposed on the output of the embedding layer to converge toward an optimal clean space, the performance of the speaker embedding is preserved for clean environments. Our proposed approaches, in all cases with different types of simulated and real noises, outperform the baseline ResNet system. For the sake of readability in the next parts of the chapter, we call the first proposed system ResNet-MSE1 and the second system named ResNet-MSE2.

7.2. PROPOSED SYSTEM

In this section, the architecture of the baseline ResNet system and the proposed variants are described.

7.2.1. BASELINE SYSTEM

The embedding extractor used in this chapter is a variant based on ResNet [46]. The ResNet model for extracting embeddings consists of three modules: a stack of *ResNet Blocks*, a *statistics-level* layer, and *segment-level* representation layer.

- ResNet (Residual Network) uses a stack of many Residual Blocks. A Residual Block is made up of two 2-dimensional Convolutional Neural Network (CNN)

layers separated by a non-linearity (ReLU). The input of the Residual Block is added to its output in order to constitute the input of the next Residual Block.

- The *statistics-level* component is an essential component to convert a variable-length speech signal into a single fixed-dimensional vector. The statistics-level is composed of one layer: the statistics pooling, which aggregates over frame-level output vectors of the DNN and computes their mean and standard deviation.
- The *segment-level* component maps the segment-level vector to speaker identities. The mean and standard deviation are concatenated together and forwarded to additional hidden layers and finally to the softmax output layer.

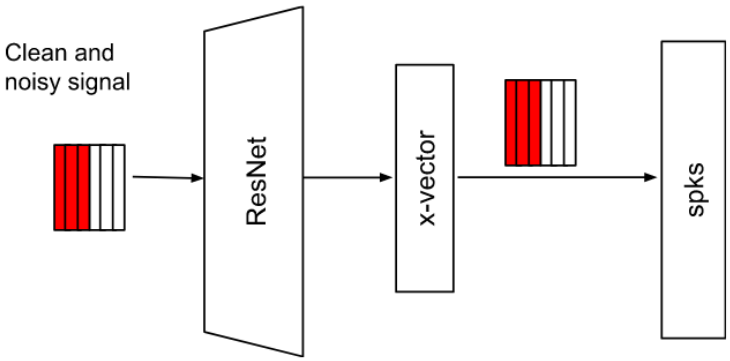


Figure 7.1: *The ResNet baseline speaker embedding extractor.*

The detailed topology of the used ResNet is shown in Table 6.1. We will show the baseline system in three blocks 7.1: the ResNet block includes all layers before the speaker embedding layer, the x-vector layer is the speaker embedding layer and the remaining layers comprise the speaker classifier.

The ResNet is trained using ArcFace softmax to classify speakers contained in the training set. Batch-norm and ReLU layers are not shown. The input is comprised of 60 filter banks from speech segments. During training, we use a fixed segment length of 400. The ResNet is trained with the Additive Angular Margin Loss (ArCFace)[21] function (Equation. 7.1):

$$\mathscr{L}_{ArcFace} = -\frac{1}{n}\sum_{i=1}^{n}\log\frac{e^{s}\cdot\left(\cos\theta_{y_i}+m\right)}{e^{s.\cos\left(\theta_{y_i}+m\right)}+\sum_{j\neq y_i}e^{s.(\cos\theta_j)}} \tag{7.1}$$

Where y_i is the ith speaker, s is a scale factor and m is the margin.

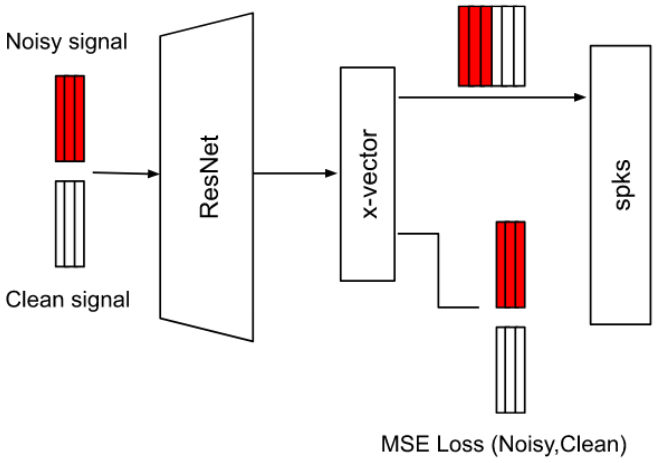


Figure 7.2: *ResNet-MSE1 generates the same representation for both and clean situations*

7.2.2. RESNET-MSE1

In this subsection, the first proposed system is described (Fig. 7.2). In the proposed system a clean signal and its corresponding noisy version are given to the network. At the embedding layer, the mean square error (MSE) between the noisy and clean speaker embedding vector is calculated at each mini-batch (Eq. 7.2):

$$\mathscr{L}_{MSE} = \sum_{i=1}^{n} \|x_i - y_i\|^2 \tag{7.2}$$

Where n is the size of the mini-batch, x_i is the ith clean speaker embedding and y_i is the ith noisy speaker embedding.

Both versions of the signal (i.e. clean and noisy) are given to the classifier. In this system a combination of ArcFace softmax and mean square error is used as the loss function (Eq. 7.3):

$$\mathscr{L}_{ResNet-MSE1} = \mathscr{L}_{ArcFace} + \lambda \mathscr{L}_{MSE} \tag{7.3}$$

When the network is updated with a noisy signal, the MSE between noisy and clean versions is reduced. In the same manner, when the network is updated on the clean version this distance is optimized for the second time. Updating the network over noisy samples improves the performance in noisy environments but training over the

clean version makes the performance of the system a little worse because the clean samples move toward the noisy sample. In the second proposed system, we resolve this problem. This behaviour is shown in Figure 7.4a. The training procedure of ResNet-MSE1 is shown in Algorithm 1.

Algorithm 1 ResNet-MSE2 training process

f : *the encoder*
cl : *speaker classifier*
N : *batch size*
(X, Y, S), *pairs of noisy and clean signal for speaker S*
for (X,Y,S) in loader **do**
  $E_X \leftarrow f(X, S)$ #clean speaker embedding
  $E_Y \leftarrow f(Y, S)$ #noisy speaker embedding
  $L_{MSE} \leftarrow \sum_{i=1}^{N} (ex_i - ey_i)^2$ #MSE error between noisy and clean embeddings
  $CE_X \leftarrow cl(E_X, S)$ #classifier error for clean signal
  $CE_Y \leftarrow cl(E_Y, S)$ #classifier error for noisy signal
  $loss \leftarrow CE_X + CE_Y + \lambda L_{MSE}$ # total loss
  $loss.backward(f, cl)$
  $optimizer.step()$
end for

7.2.3. RESNET-MSE2

In the second proposed system, the performance degradation in clean environments is resolved. To do that we used an assistance pre-trained network which is similar to our baseline system. Firstly a speaker embedding network with a mixture of clean and noisy data is trained. The pre-trained network is used to extract speaker embeddings for a clean version of the training dataset. We assume these vectors as the best version we can achieve. After that, another network is used to converge the noisy version of speaker embeddings towards the clean version extracted in the previous step. In this step, the system is trained with pairs of noisy and clean versions. At the speaker embedding layer, the MSE between the fixed clean version and a given training sample (for both clean and noisy versions) is calculated.

$$\mathscr{L}_{MSE2} = \sum_{i=1,p=1}^{n} \|x_p - y_i\|^2 \tag{7.4}$$

Where n is the size of the mini-batch, x_p is the pth clean speaker embedding extracted from the pre-trained network, and y_i is the ith noisy speaker embedding.

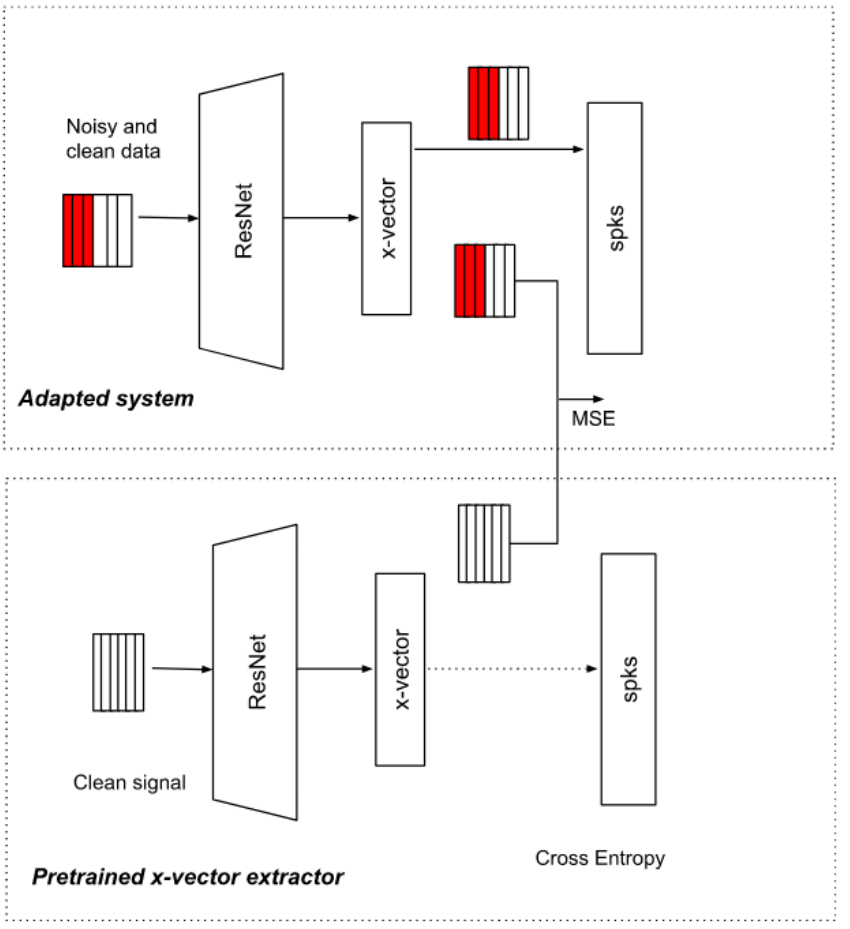


Figure 7.3: ResNet-MSE2 impose on the noisy embedding to have the same representation as the best clean embedding

Both clean and noisy versions are given to the classifier and the weights are updated with all samples. In this system a combination of ArcFace softmax and mean square error is used as the loss function (Eq. 7.5):

$$L_{ResNet-MSE2} = L_{ArcFace} + \lambda L_{MSE2} \tag{7.5}$$

This procedure is depicted in Figure 7.3. The steps of convergence towards the clean speaker embedding are shown in Figure 7.4b. The training procedure of ResNet-MSE2 is shown in Algorithm 2. Both ResNet-MSE1 and ResNet-MSE2 are implemented in Pytorch and stklia.

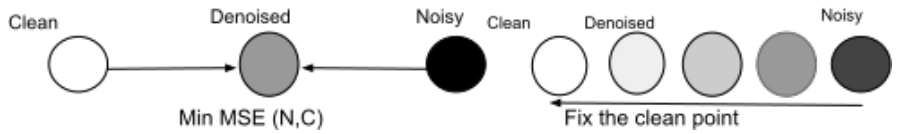


Figure 7.4: The impact of applying MSE in the proposed systems Left) ResNet-MSE1 generates identical representations for both clean and noisy situations Right) It is imposed on the noisy embedding to have the same representation as the best clean embedding

7.2.4. EXPERIMENTAL SETUP

Algorithm 2 ResNet-MSE2 training algorithm

f_C : *the pretrained encoder* f_N : *the noisy encoder*
cl_N : *speaker slassifier*
N : *Batch size*
(X, Y, S), *pairs of noisy and clean signal for speaker* S
for (X,Y,S) in loader **do**
$E_X \leftarrow f_C(X, S)$ #clean speaker embedding
$E_Y \leftarrow f_N(Y, S)$ #noisy speaker embedding
$L_{MSE2} \leftarrow \sum_{i=1,p=1}^{N} (ex_p - ey_i)^2$ #MSE error between pre-trained clean embedding and target network embedding
$CE_X \leftarrow cl(E_X, S)$ #classifier error for clean signal
$CE_Y \leftarrow cl(E_Y, S)$ #classifier error for noisy signal
$loss \leftarrow CE_X + CE_Y + \lambda L_{MSE}$ # total loss
$loss.backward(f_N, cl_N)$
$optimizer.step()$
end for

7.2.5. SPEAKER EMBEDDING EXTRACTORS

All speaker embedding extractors are trained with Voxcleeb2 in 10.000 iterations. The learning rate at the beginning of the training is set to 0.2 and weight decay equals to 2.10^{-4}. The momentum is set to 0.9. In all experiments, the stochastic gradient descent optimizer is used. The size of the feature maps is 32, 64, 128, and 256 for the 4 ResNet blocks.

- **Baseline**. In the baseline ResNet, the training samples are chosen from clean Voxceleb randomly. The training data includes all clean files of Voxceleb and their augmented version with MUSAN Corpus and reverberated with a pool of

RIR files [1]. Kaldi toolkit is used for data augmentation [29]. The batch size is set to 128.

- **ResNet-MSE1**. In this system, a clean file from Voxceleb was chosen randomly. After that its augmented version was chosen. Because we have two versions of each file at each mini-batch, we reduced the size of each mini-batch to 64.
- **ResNet-MSE2**. In this system at each mini-batch, a speaker embedding extracted from the baseline system was chosen. At the same time, the clean or noisy signal of the chosen files is selected. The modified system tries to reduce the distance between the clean speaker embedding and the speaker embedding extracted from the given signal.

7.2.6. TEST PROTOCOLS

- **Fabiole1**. In the first protocol, the Fabiole corpus is used. In this protocol 130 (one file per speaker) are used as enrollment and 1870 randomly chosen files are used for the test. In this protocol, the BBC noise files are used as artificial noise and data simulation is done with Kaldi.
- **Fabiole2**. The test and enrollment files in this protocol are the same as the previous one. But in this protocol, we used Freesound noises that are completely different from the noises used in the training data. In this experiment, we used Pyroomacoustics[2] for data simulation
- **Robovox**. In this protocol, 26 files, one file per speaker are used as the enrollment, and 677 files are used as the test. The enrollment files are chosen from a closed microphone with high quality but the test files are chosen from a far microphone between 1 and 3 meters.
- **VoiCes**. In this protocol, 300 files, one file per speaker are used as enrollment, and 300 files are used as the test. The enrollment files adopted from the Librispeech (the clean version of Voices) and the test files are the replayed files in VoiCes recorded in room 3 and room 4 in the presence of severe music noise, and reverberation. We used mic 5, which is the farthest microphone in Voices.

The details of all protocols are summarized in Table 7.1.

[1] http://www.openslr.org/resources/28/rirs_noises.zip
[2] https://github.com/timmahrt/pyroomcoustics

Table 7.1: *Test protocols.*

Protocols	Test	Enroll	Trials
Fabiole1	1870	130	243k
Fabiole2	1870	130	243k
Robovox	26	677	17k
Voices	300	300	90k

7.2.7. RESULTS AND DISCUSSION

In this section, the obtained results are discussed. Our results show that in all cases our proposed systems are more robust in noisy environments. However, the performance of the first proposed system reduces in clean environments in comparison to the baseline system, the performance of the second system for clean environments remains stable.

In Table 7.2 the results for the Fabiole1 protocol are presented. As is shown, the modified systems improve significantly in comparison to the baseline system. The first column shows the results for clean environments and the other columns include the results with different SNRs.For example, when SNR is between 0 and 5 the EER is 14 % with the second system.

Table 7.2: *Fabiole 1 protocol results with baseline ResNet and proposed variants (EER).*

System	Clean	Noisy 0-5	Noisy 5-10	Noisy 10-15
Baseline	5.20	7.96	7.43	7.00
ResNet-MSE1	5.40	7.43	6.79	6.09
ResNet-MSE2	**5.19**	**6.79**	**5.98**	**5.66**

In the Fabiole2 protocol, we showed the generalizability of proposed systems to other noises and RIR simulators. In this protocol, the same test and enrollment files as the Fabiole1 protocol are used. But the Freesound noises dataset and Pyroomacoustics library are used for data simulation. Table 7.3 shows that in all cases the proposed systems reduce the impact of noise and reverberation. It deserved to be mentioned that in this protocol the results for the clean situation are worse in comparison to the Fabiole1 protocol because test files are truncated to 15 seconds for both clean and noisy situations.

In order to extend the capability of the proposed systems to real environments. The experiments are done on the Robovox dataset in Table 7.4. The first column shows the

Table 7.3: *Fabiole 2 protocol results with baseline ResNet and proposed variants (EER).*

System	Clean	Noisy 0-5
Baseline	7.11	12.19
ResNet-MSE1	7.30	11.18
ResNet-MSE2	**6.95**	**10.46**

results with the best channel with a closed microphone. As it is shown in the Fabiole protocol the ResNet-MSE1 system is worse in a clean environment in comparison to the baseline system and the results for the ResNet-MSE2 system are the same as the baseline system. In the third column, the results are shown for far microphone in the presence of noise and reverberation. Both adapted systems give better results in comparison to the baseline system. The last column shows the results with simulated noise in the Robovox protocol. In this experiment, the clean data from channel 3 is augmented with Freesound noises and Pyroomacosutics with SNR between 0 and 5. In this experiment, the behavior of the proposed systems is the same. For real noise the relative improvement with the ResNet-MSE2 system is 15% and for simulated noise the gain is 10%.

Table 7.4: *Robovox protocol results with baseline ResNet and proposed variants(EER).*

System	Ch5	Ch3	Noisy 0-5
Baseline	2.21	4.38	6.59
ResNet-MSE1	2.36	4.10	6.05
ResNet-MSE2	**2.21**	**3.69**	**5.90**

Finally, the systems are tested with a protocol created from the VoiCes dataset. The results are shown in Table 7.5. The experiments show that with the ResNet-MSE2 system, there is 5% relative improvement of EER.

Table 7.5: *Voices protocol results with baseline ResNet and proposed variants(EER).*

System	Clean	Room 4 Music
Baseline	0.66	6.33
ResNet-MSE1	0.66	6.33
ResNet-MSE2	0.66	**6.00**

The obtained results show the feasibility of doing noise compensation at the

speaker embedding extractor level. The results obtained from ResNet-MSE1 in a clean environment show that reducing the distance between noisy and clean environments without fixing the clean point makes the system worse in clean environments because it converges to a space between the noisy and clean spaces. To solve this problem we fixed the clean speaker embedding in the second proposed system. Also, the results show that the second system is superior in noisy environments because during the process of reducing the distance between noisy and clean speaker embeddings the noisy speaker embedding moves closer toward the clean point to minimize the MSE. The MSE between noisy and clean speaker embeddings are shown in Table 7.6.

Table 7.6: MSE distance between pairs of noisy-clean speaker embeddings before and after optimizing MSE with CE loss.

System	Fabiole1	Fabiole2	Robovox	VoiCes
Baseline	2.85	7.14	4.05	10.23
ResNet-MSE1	0.06	0.015	0.09	0.02
ResNet-MSE2	0.66	1.71	1.01	2.67

7.3. CONCLUSION

In this chapter, we introduced two variants of the ResNet-based speaker recognition system to integrate a noise compensation module with the speaker embedding in order to make the speaker embedding more robust against additive noise and reverberation. In the first system, the network is updated to reduce the distance between pairs of noisy/clean speaker embeddings in the embedding layer. In the second system, an optimal clean point is fixed and at each iteration, the noisy and clean speaker embeddings given by the signal in the input are shifted toward the optimal point. Training speaker embedding extractors with the proposed strategies makes the speaker embedding more robust against additive noise and reverberation.

8

BARLOW TWINS SELF-SUPERVISED LEARNING FOR ROBUST SPEAKER RECOGNITION

The general improvement of speaker embeddings by capturing more speaker discriminant information and their robustness against specific acoustical variabilities are the main approaches that have been taken into account to achieve better systems. In the previous chapters, we explored the robustness of speaker recognition systems against noise and reverberation. In this chapter, we will propose a self-supervised speaker embedding approach that tries to improve the performance of speaker recognition systems in general and in the presence of noise and reverberation. Our proposed approach is based on Barlow Twins self-supervised loss function. Barlow Twins objective function tries to optimize two criteria: Firstly, it increases the similarity between two versions of the same signal (i.e. the clean and its augmented noisy version) to make the speaker embedding invariant to the acoustic noise. Secondly, it reduces the redundancy between the dimensions of the speaker embeddings which improves their discriminability in general. In our research, the Barlow Twins objective function is integrated with the ResNet-based speaker embedding system. In the proposed system, the Barlow Twins objective function is calculated in the embedding layer and it is optimized jointly with the speaker classifier loss function. The experimental results on the Fabiole corpus show a 22% relative gain in terms of EER in clean environments and an 18% improvement in the presence of noise with low SNR and reverberation.

The presented work in this chapter is published in [134]

8.0.1. INTRODUCTION

The speaker embeddings encode both speaker discriminant features and secondary features such as transmission channel, acoustical variability, and content information. Also, there is redundant information distributed over the dimensions of embeddings. The acoustical variabilities and redundant information reduce the discriminability of speaker embeddings. Discarding redundant information improves the quality of speaker embeddings in general and for all situations. Suppressing the variabilities' negative impact makes the speaker recognition systems more robust for specific situations such as the presence of noise and reverberation. In the previous chapters, we worked particularly on noise and reverberation compensation in TDNN and ResNet speaker recognition systems. In this chapter, we go further to improve the performance of ResNet-based speaker recognition systems for both clean and noisy environments.

Although the DNN-based speaker embedding systems have given a degree of robustness against acoustic noises, there is a significant degradation of their performance in the presence of background noise, reverberation, and other variabilities [53] [126] [127]. Various approaches have been proposed to handle these variabilities in different parts of the system such as signal level [72], feature level [82], speaker modeling level [135], speaker embedding level [127] and scoring technique level [136]. Addressing the variabilities at each step has its own advantage and disadvantages in terms of data, computational resources, efficiency, etc. In this chapter, we chose to make the ResNet-based speaker embedding extractor more robust against background noise and reverberation with a self-supervised objective function named Barlow Twins [137]. In the current work, we worked on the speaker modeling level. Because reducing the impact of noise and reverberation in higher levels is limited [132], having a noise-robust speaker embedding system is highly demanded.

The goal of self-supervised learning (SSL) is to learn a robust and invariant representation of the same data samples in the presence of different distortions (i.e. additive noise and reverberation in our case). Several self-supervised learning methods are proposed for robust data representation [138] [139]. Although the contrastive loss function has given promising results in domain adaption and robust speaker recognition, it has some limitations such as the necessity for large batch size and the way of defining the negative pairs [140].

Some self-supervised learning methods are used for domain adaptation in speaker recognition systems. In a self-supervised approach, [120] the softmax loss function is trained with the contrastive loss. Because joint training of softmax loss function and contrastive function is intricate, they optimized the contrastive loss to fine-tune

a network that is trained with a softmax loss function. In [97] a domain adaptation technique is proposed that uses mean discrepancy distance (MMD) as a regularizer integrated with speaker embedding that performs the adaptation between the source and target domain.

In this chapter, we introduce the Barlow Twin objective function in the domain of speaker recognition systems. Barlow Twins is a self-supervised objective function that has two goals. Firstly, it increases the similarity between two versions of the same signal to give an invariant representation. Secondly, it reduces the redundancy between different dimensions of speaker embeddings. The first goal makes the speaker representations more robust against variabilities and the second goal improves the discriminability of representations that improves the overall quality of the representations [137] [141]. The Barlow Twins objective function is integrated with the ResNet-based speaker embedding system. In the proposed system, the Barlow Twins objective function is jointly optimized with the Additive Angular Margin Loss function. The Additive Angular Margin Loss is obtained from the last layer of the ResNet system that classifies the speakers and the Barlow Twins function is calculated over the clean version and its noisy corresponding version of speaker embeddings extracted at the embedding layer of the ResNet network at each mini-batch.

8.0.2. Proposed approach

Baseline system

The baseline speaker embedding extractor used in this chapter is a variant based on ResNet [46]. The ResNet model for extracting embeddings consists of three modules: a set of *ResNet Blocks*, a *statistics-level* layer, and *segment-level* representation layers. The detailed topology of the used ResNet is shown in Table 6.1. The ResNet is trained with 60 Mel scale filter banks from speech segments. During training, we use a fixed segment length of 400 frames equals 4 seconds. The ResNet system is trained with the Additive Angular Margin Loss function.

Barlow Twins system

In the Barlow Twins system, the baseline ResNet network accepts the clean version and its augmented version of the clean signal at each mini-batch. Therefore, in the embedding layer, we have pairs of clean and noisy embeddings that fed into the speaker classifier and the Barlow Twins objective function. The architecture of the proposed system is depicted in Fig. 8.1. The generator accepting noisy and clean signals are identical.

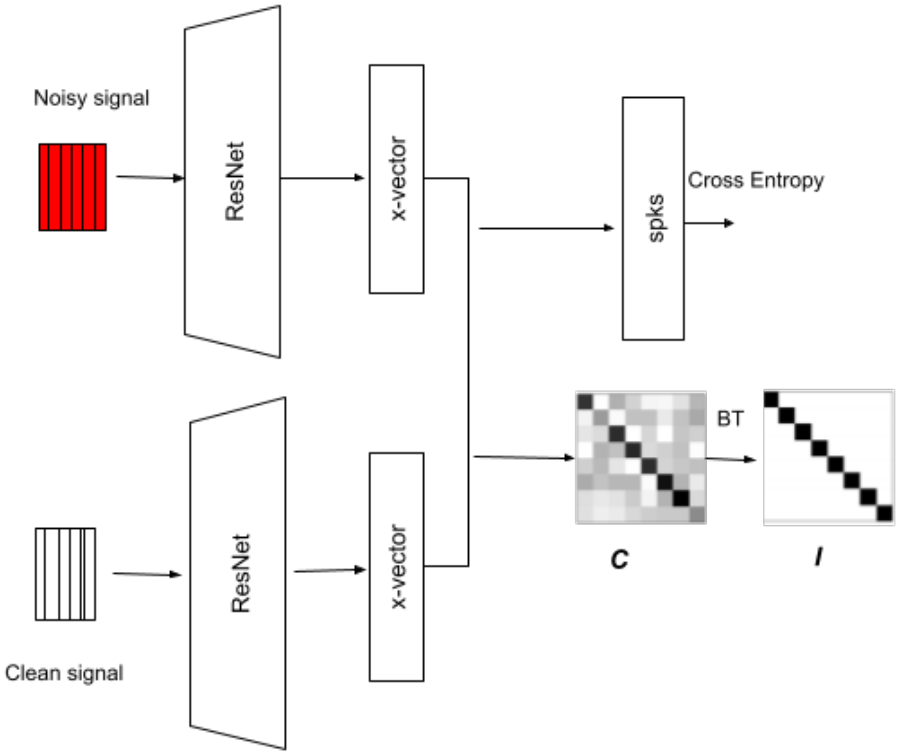


Figure 8.1: The configuration of Barlow Twins robust speaker recognition system

The Barlow Twins objective functions accept two sets of inputs: z^X and z^Y are the mean-centered normalized versions of clean and noisy speaker embeddings obtained from the embedding layer of the ResNet system at each mini-batch.

The Barlow Twins function is defined as Equation.8.1:

$$L_{BT} = \sum_{i}(1 - C_{ii})^2 + \lambda \sum_{i!=j}(C_{ij})^2 \tag{8.1}$$

where C is the correlation matrix between the output of noisy and clean speaker embeddings at each mini-batch and, i and j are speaker embedding dimensions. The C matrix is a square matrix; its dimension is equal to the size of the speaker embedding. The first term is called invariant term which tries to increase the correlation between two versions of noisy and clean speaker embeddings, in order to give an invariant speaker embedding for noisy and clean versions, the second term is

called redundancy reduction which reduce the redundancy within the dimensions of the embedding, the λ coefficient indicates the importance of each term, and C_{ij} is defined as Equation.8.2:

$$C_{ij} = \frac{\sum_b z^X_{b,i} z^Y_{b,j}}{\sqrt{\sum_b (z^X_{b,i})^2} \sqrt{\sum_b (z^Y_{b,j})^2}} \tag{8.2}$$

the sum is performed over all the embeddings in a given mini-batch, and b is the index of an embedding in the mini-batch.

In the proposed system, the Barlow Twins objective function is optimized jointly with the Additive Angular Margin Loss (function that is used in the speaker classifier. The final objective function is the summation of Additive Angular Margin Loss (ArcFace) [142] loss and Barlow Twins objective function. Both functions are optimized with the same weight. In the end, the Barlow Twins objective function imposes on the correlation matrix to be an identity matrix (it maps the C matrix to I identity matrix shown in Fig. 8.1). The optimization of Barlow Twins brings the diagonal elements closer to 1 to have the invariant representations and imposes on off-diagonal elements of the correlation matrix to be close to 0. The training procedure of the proposed system is shown in Algorithm 3.

Algorithm 3 Barlow Twins self-supervised learning algorithm

f_C: *the clean encoder* f_C: *thenoisy encoder*
cl_N: *speaker classifier*
b: *batch size*
(X, Y, S), *pairs of noisy and clean signal for speaker* S
for (X,Y,S) in loader **do**
  $E_X \leftarrow f_C(X, S)$ #clean speaker embedding
  $E_Y \leftarrow f_N(Y, S)$ #noisy speaker embedding
  $z^X = (E_X - E_X.mean(0))/E_X.std(0)$ #NxD Normalized clean embeddings
  $z^Y = (E_Y - E_Y.mean(0))/E_Y.std(0)$ #NxD Normalized noisy embeddings
  $C_{ij} = \frac{\sum_b z^X_{b,i} z^Y_{b,j}}{\sqrt{\sum_b (z^X_{b,i})^2} \sqrt{\sum_b (z^Y_{b,j})^2}}$ #DxD Correlation matrix
  $L_{BT} = \sum_i (1 - C_{ii})^2 + \lambda \sum_{i!=j} (C_{ij})^2$ # Barlow Twins loss value
  $CE_X \leftarrow cl(E_X, S)$ #classifier error for clean signal
  $CE_Y \leftarrow cl(E_Y, S)$ #classifier error for noisy signal
  $loss \leftarrow CE_X + CE_Y + \gamma L_{BT}(E_X, E_Y)$ # total loss
  $loss.backward(f_N, cl_N)$
  $optimizer.step()$
end for

Considering that a mini-batch is a matrix of D rows and B columns; where B is the size of the mini-batch and D is the size of an embedding, we can see that C_{ij} is the cosine between the vector lines of indices i ($z^X_{b,i}$) and j ($z^Y_{b,j}$) of the mini-batch. The C_{ii} is calculated over the same dimension of speaker embeddings in the clean and noisy mini-batch and C_{ij} is calculated over the ith dimension of speaker embeddings in the clean mini-batch and jth dimension noisy mini-batch or vice versa. This process is shown in Figure 8.2. D_S and D_T stand for pairs of speaker embeddings in clean and noisy environments respectively.

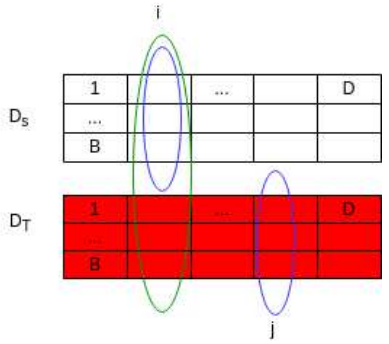


Figure 8.2: Barlow Twins calculation procedure on pairs of noisy and clean speaker embedding at mini-batch

The Barlow Twins objective function minimizes the distances between row vectors with the same indices and maximizes the distances between row vectors having different indices. In this sense, it is similar to the contrastive learning objective function. Indeed, it is the same formulation applied to the lines of the mini-batch in the case of the Barlow Twins and applied to the columns of the mini-batch in the case of contrastive learning. In the calculation of invariant terms in Barlow Twins, each element comes from different speakers with different noises. It means that both noise and speaker variabilities are present. While in calculating the distance between the positive pairs in contrastive loss only the noise variability is taken into account because two samples come from the same speaker that is augmented with different noises. The Barlow Twins system is implemented in Pytorch.

8.0.3. EXPERIMENTS SETUP

In this section, the experiment's setup including the used datasets, speaker embedding extractors, and evaluation protocols are described.

SPEAKER EMBEDDING EXTRACTORS

Both baseline and Barlow Twins speaker embedding extractors are trained with Voxcleeb in 10,000 iterations. In another experiment, the Barlow Twins were used with a pre-trained baseline system. In the last case, the Barlow Twins and Additive Angular Margin Loss function are optimized together for 1,000 more iterations of the baseline system. The learning rate at the beginning of the training is set to 0.2 with weight decay equals 2.10^{-4}. The momentum is set to 0.9. The gradient descent optimizer is used. The size of the feature maps is 32, 64, 128, and 256 for the 4 ResNet blocks.

- **Baseline**. In the baseline system, the training samples are chosen randomly. The training data includes all clean files of Voxceleb and their augmented version with Musan Corpus and reverberated with a pool of RIR files. [1]. Kaldi toolkit is used for data augmentation [29]. The SNR was chosen between 0 and 20. The batch size is set to 128. In this system, only the Additive Angular Margin Loss function is optimized.
- **Barlow Twins**. In this system, a clean file from Voxceleb was chosen randomly. After that, its augmented version was chosen. Because we have two versions of each file at each mini-batch, we reduced the size of each mini-batch to 64. At the embedding layer, the Barlow Twins objective function is calculated and the proposed system was updated to minimize the summation of Barlow Twins and Additive Angular Margin Loss functions. The λ variable is set to 0.005 in Equation. 8.1. The λ is chosen experimentally. The large values for λ cause gradient explosion.

8.0.4. EVALUATION PROTOCOLS

- **Fabiole**. In the first protocol, the Fabiole corpus is used. In this protocol 130 files (one file per speaker) are used as enrollment and 6,870 randomly chosen files are used for the test. In this protocol, the BBC noise files are added to the clean signal with different SNRs from 0 to 15. In all cases, the clean signal is used for enrollment. In this protocol, the Kaldi toolkit is used to add noises to clean files.
- **Robovox**. In this protocol, 26 files, one file per speaker, are used as the enrollment, and 677 files are used as the test. The enrollment files are chosen from a closed microphone with high quality but the test files are chosen from far microphones. The average length of speech utterances in this protocol is 22 seconds.

[1] http://www.openslr.org/resources/28/rirs_noises.zip

The details of both protocols are summarized in Table 8.1.

Table 8.1: Experimental Protocols, designed on Fabiole and Robovox corpora

Protocol	Trials	Test	Enrolment
Fabiole	893k	6870	130
Robovox	17k	677	26

8.0.5. RESULTS AND DISCUSSIONS

In this section, the obtained results are discussed. The results obtained from the Fabiole protocol are presented in Table 8.2. The BT column shows the results for a system in which Barlow Twins is optimized from scratch with the speaker classifier. For example in a clean environment, EER reduces from 6.27 to 4.87 which means 22% relative gain. In the case of low SNR between 0 and 5, we achieved an 18% relative gain of EER. The results for a case where Barlow Twins were used with a pretrained baseline system are presented in the last column. In this experiment, in all cases, we observed significant improvement of EER but the results for the training of Barlow Twins from scratch are better.

Table 8.2: Fabiole Protocol results obtained by joint optimization of speaker classification loss and Barlow Twins (EER)

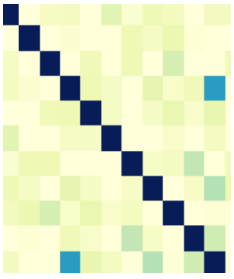
SNR	Baseline	BT	Pre+BT
Clean	6.27	**4.87**	5.46
[0-5]	8.31	**6.81**	7.37
[5-10]	7.43	**5.86**	6.66
[10-15]	6.87	**5.48**	6.17

The results with the Robovox protocol are presented in Table 8.3. In the BT column, the results are shown for the case of joint optimization of both loss functions from scratch. In this experiment, we observed significant improvement for some channels but the behavior is not the same in all channels. The obtained results show that in the clean situation (i.e. channel 5) the Barlow Twins improve the performance. In other channels that are far and noisy, the results are paradoxical. Finally, the results of an experiment which Barlow Twins adapts the pre-trained baseline system are presented in the last column. In this case, we observed improvement in all channels for example in the clean environment there is a 33% relative gain.

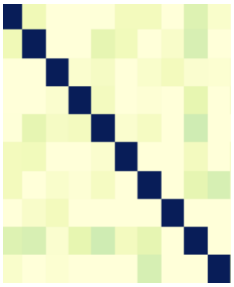
Table 8.3: Robovox Protocol results obtained by joint optimization of speaker classification loss and Barlow Twins(EER)

Channel	Baseline	BT	Pre+BT
1	4.28	4.28	**3.98**
2	4.72	4.43	**4.13**
3	4.38	4.57	**3.84**
4	5.76	5.90	**5.31**
5	2.21	1.92	**1.47**

In order to show the impact of Barlow Twins on decorrelating the speaker embedding dimension we will do a visualization of the correlation matrix. Figure 8.3 shows the correlation matrix between noisy and clean speaker embeddings extracted by the baseline system and Barlow Twins systems. We can see that the Barlow Twins decorrelated the dimensions partially. It is expected to have more decorrelated dimensions with a bigger coefficient of redundancy reduction term but our experiments show that giving bigger weight to this term cause gradient explosion.



(a) Normal ResNet



(a) Barlow Twins

Figure 8.3: The correlation matrix for speaker embedding dimensions before and after optimizing Barlow Twins

In another experiment, the impact of different values for λ hyperparameter is explored. The results are presented in Table 8.4.

Table 8.4: Applying the Barlow Twins with different values of $\lambda$for redundancy term

Ch/λ	**0.5**	**0.05**	**0.005**
1	4.57	**3.84**	3.98
2	4.43	**4.13**	**4.13**
3	4.57	3.98	**3.84**
4	6.2	**4.72**	5.31
5	1.92	1.77	**1.47**

The results show that for $\lambda = 0.5$ the system gives the worst results. From the training process, we observed that bigger values for λ make the optimization of speaker classification loss more difficult which results in low performance. In the case of $\lambda = 0.05$ the improvement for channel 4 is significant which shows that for noisy situations there is more redundant information and increasing the weight of the Barlow Twins loss function helps to remove this unwanted information encoded in the speaker embedding dimensions.

8.0.6. CONCLUSION

In this chapter, the Barlow Twins objective function was introduced in the area of robust speaker recognition systems. The Barlow Twins objective function integrated with ResNet speaker embedding in order to achieve two goals: give an invariant representation for both clean and noisy versions of a speech signal, and reduce redundancy between different dimensions of the speaker embedding. We showed that the Barlow Twins objective function improves the performance of the speaker embedding system in both noisy and clean environments. The joint optimization of contrastive loss and Barlow Twins loss function in robust speaker recognition is a potential future work. The proposed system improves speaker recognition performance for both noise-free and noisy environments. Also, it can generalize well in the case of real noise and reverberation. In future work, the behavior of the Barlow Twins objective function in the presence of specific noises and with more data augmentation techniques will be studied.

9

CONCLUSION

Speaker recognition systems authenticate the identity of claimed users from their speech utterances. The authentication is done by comparing the fixed-length representations of enrollment (i.e. registered user) with test files (i.e. claimed user). The performance of speaker recognition systems is highly impacted by the quality of these representations. However, it is possible to have clean and enough speech data for enrollment, the system is used in various environments where there are different interferences that distort the speech signal. The distorted speech causes less speaker-discriminant representations that result in the performance degradation of the speaker recognition system.

The state-of-the-art speaker recognition systems based on DNNs are more robust against acoustical distortions in comparison to their precedent statistical generation. Despite this relative robustness severe acoustical distortion reduce the performance of DNN-based speaker recognition systems.

In the first part of this thesis, the DNN-based speaker recognition systems and their challenges have been discussed. In Chapter 1, a general taxonomy of speaker recognition systems was presented and their challenges were discussed. We posited the place of studied systems in this thesis as the robustness of text-independent open-set speaker recognition systems against additive noise and reverberation.

In Chapter 2 the main DNN-based speaker recognition systems including TDNN, ResNet, MFA-Conformer, and ECAPA TDNN are reviewed. In addition to the architecture of speaker embedding DNNs, the pooling strategies, objectives function, and scoring methods are discussed. The characteristics of different parts of speaker recognition systems are important to archive a more robust pipeline.

In Chapter 3 the proposed strategies used for making speaker recognition

robust against additive noise and reverberation are discussed. The speech enhancement methods including masking-based, mapping-based, and adversarial training approaches reviewed. Another common approach is training a robust speaker embedding extractor. This approach can be applied to a bigger number of variabilities including content variabilities in comparison to speech enhancement. The speaker embedding transformation is the third group of reviewed robust techniques. The reviewed literature shows that noise compensation at the speaker embedding level is more promising in comparison to other approaches.

Three main reasons motivated us to study the robustness of speaker recognition systems at the speakers modeling level and speaker embeddings. First of all the improvement of speech quality in speech enhancement methods doesn't guarantee the improvement of the speaker recognition systems. Secondly, the achieved results in the speaker embedding level are more substantial. Last but not least is the low-computational cost and the facility of adapting the compensation module for specific situations. In Chapter 5 a general framework of noise compensation at the x-vector level has been proposed. The contributing findings are:

- The TDNN-based speaker recognition systems are not robust against additive noise in low SNR and reverberation.
- Adding a noise compensation module before scoring can compensate for the negative impact of noise and reverberation significantly.
- Several noise compensation modules based on DAEs proposed. The Stacked DAE is the most successful one.
- Both statistical and different variants of DAE can compensate for additive noise.
- Data augmentation and increasing the number of speakers is crucial to make the speaker recognition robust against noise, but it is not sufficient and still, the noise compensation module can help us to achieve better results.
- Since the DAEs learn a hyperplane to make the noisy x-vector close to their clean distribution, we don't need to have specific noise compensation modules for different noises and a general model can do the compensation for different kinds of noises
- In the case of having more than one distortion two configurations proposed. Similar to specific noise compensation, we observed that we don't need to have a specific noise compensation module for each distortion.

In Chapter 6 the behavior of the ResNet speaker recognition system facing noise and reverberation has been investigated. The obtained results are compared with the TDNN system. We found:

- Noise compensation at speaker embedding level with ResNet system brings a marginal improvement in terms of EER.
- Even without noise compensation, the performance of the ResNet system is higher in comparison to TDNN with noise compensation.
- The noise compensation module doesn't bring a significant improvement in facing real noise and reverberation in both TDNN and ResNet systems. Despite our endeavor to find an efficient training data simulation and fine-tune environmental parameters that impact noise and reverberation, we found that having a general rich dataset that considers all kinds of environments is more plausible with noise compensation techniques.

Because of the limitation found in noise compensation, we shifted to improving the speaker embedding network. In Chapter 7 we proposed two noise-robust learning methods with the ResNet system. Our contributions are:

- We proposed a noise and reverberation invariant system that learns the same distribution for both noisy and clean environments. This system gives better results for noisy environments but the results for clean situations are worse.
- We proposed a second noise invariant system that imposes on the speaker embeddings for noisy environment shift towards the distribution of the best-obtained system in a clean environment.
- The experimental results obtained by our systems show a significant improvement of the ResNet system against both artificial and real noise and reverberation.

In Chapter 8 we proposed a self-supervised learning framework based on Barlow Twins loss function to make the speaker embeddings invariant to noise and reverberation and decrease the redundancy between speaker embedding dimensions. The proposed approach imposes on the speaker embedding network to improve the quality of speaker embeddings in two ways.

- A redundancy reduction term in Barlow Twins decorrelate different dimensions of speaker embeddings in order to increase the discriminability of embeddings
- A noise invariant term imposes on a different augmented version of embeddings to be invariant to noise and reverberation distortions.
- The experiments show the effectiveness of both decorrelation and noise invariant strategies and the proposed system gives better results for both clean and noisy environments.

9.1. PERSPECTIVES AND FUTURE WORK

Noise compensation at the speaker embedding level can be extended in several directions. Firstly, the noise compensation can be extended to other speaker embedding extractors such as MFA-Conformer and ECAPA-TDNN. Noise compensation with these systems can give a broader insight into the efficiency of this methodology and the behavior of these systems at the speaker embedding level.

The ResNet and ECAPA-TDNN are equipped with variants of angular softmax objective functions such as the speaker classifier loss function that redistribute the output classes over a more complex manifold. One possible reason behind the marginal improvement of noise compensation in the ResNet system is using the angular objective function. We propose to do noise compensation with a denoising autoencoder that uses a similar approach as angular softmax.

However, our experimental results show that using a highly diversified data augmentation method generalizes better in training noise compensation modules, it is important to use other proposed data augmentation techniques such as Rawboost, or heuristic-based optimal augmentation. Because such methods can simulate other variabilities such as transmission channels.

The hand-made MFCC and filter banks are the used features throughout our thesis, we propose to explore the recent unsupervised speech representation methods such as WavLM that are more robust against additive noise and reverberation.

Our work in Chapter 7 can be extended in several ways. In our experiments, we used the MSE loss function to generate noise-invariant representations. We proposed replacing MSE with more flexible regularizers such as MMD [97] or Coral [94]. Also, this system can be extended to an adversarial framework by replacing the regulator with a discriminator. Furthermore, it is possible to use the potential of both regularisation and adversarial training.

The proposed system in chapter 8 can be explored in more detail. The Barlow Twins were improved in speaker embedding level. Since Barlow twins are optimized over the speaker embedding dimension, we propose to calculate and optimize it in the pooling layer where the number of dimensions is several times higher than the speaker embedding layer. Also, we propose to optimize it hierarchically in frame-level layers.

Finally, we propose to combine Barlow Twins with other self-supervised approaches such as contrastive learning. The Barlow Twins redundancy reduction term improves the performance of the system significantly. Although, the lack of necessity for negative pairs is a positive characteristic of Barlow twins, in the case of additives noise

and reverberation variabilities it is possible to produce several images for a given signal that can be used as negative pairs. In such cases, using a contrastive loss can assist in generating the equivalent representations for the different distorted versions of a speaker embedding.

BIBLIOGRAPHY

[1] E. Variani, X. Lei, E. McDermott, I. L. Moreno, and J. Gonzalez-Dominguez. "Deep neural networks for small footprint text-dependent speaker verification". In: *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2014, pp. 4052–4056. DOI: 10.1109/ICASSP.2014.6854363.

[2] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur. "X-Vectors: Robust DNN Embeddings for Speaker Recognition". In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2018, pp. 5329–5333. DOI: 10.1109/ICASSP.2018.8461375.

[3] H. Zeinali, S. Wang, A. Silnova, P. Matjka, and O. Plchot. *BUT System Description to VoxCeleb Speaker Recognition Challenge 2019*. 2019. DOI: 10.48550/ARXIV.1910.12592. URL: https://arxiv.org/abs/1910.12592.

[4] Y. Zhang, Z. Lv, H. Wu, S. Zhang, P. Hu, Z. Wu, H.-y. Lee, and H. Meng. "MFA-Conformer: Multi-scale Feature Aggregation Conformer for Automatic Speaker Verification". In: *Proc. Interspeech 2022*. 2022, pp. 306–310. DOI: 10.21437/Interspeech.2022-563.

[5] B. Desplanques, J. Thienpondt, and K. Demuynck. "ECAPA-TDNN: Emphasized Channel Attention, Propagation and Aggregation in TDNN Based Speaker Verification". In: *Proc. Interspeech 2020*. 2020, pp. 3830–3834. DOI: 10.21437/Interspeech.2020-2650.

[6] C. Richey, M. A. Barrios, Z. Armstrong, C. Bartels, H. Franco, M. Graciarena, A. Lawson, M. K. Nandwana, A. Stauffer, J. van Hout, P. Gamble, J. Hetherly, C. Stephenson, and K. Ni. "Voices Obscured in Complex Environmental Settings (VOiCES) Corpus". In: *Proc. Interspeech 2018*. 2018, pp. 1566–1570. DOI: 10.21437/Interspeech.2018-1454.

[7] A. Brown, J. Huh, J. S. Chung, A. Nagrani, D. Garcia-Romero, and A. Zisserman. *VoxSRC 2021: The Third VoxCeleb Speaker Recognition Challenge*. 2022. DOI: 10.48550/ARXIV.2201.04583. URL: https://arxiv.org/abs/2201.04583.

[8] O. Sadjadi, C. Greenberg, E. Singer, L. Mason, and D. Reynolds. *NIST 2021 Speaker Recognition Evaluation Plan*. en. 2021-07-12 04:07:00 2021. URL: https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=932697.

[9] M. Rouvier and M. Mohammadamini. "Far-Field Speaker Recognition Benchmark Derived From The DiPCo Corpus". In: *Proceedings of the Thirteenth Language Resources and Evaluation Conference*. Marseille, France: European Language Resources Association, June 2022, pp. 1955–1959. URL: https://aclanthology.org/2022.lrec-1.209.

[10] S. Mdhaffar, J.-F. Bonastre, M. Tommasi, N. Tomashenko, and Y. Estève. "Retrieving Speaker Information from Personalized Acoustic Models for Speech Recognition". In: *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2022, pp. 6767–6771. DOI: 10.1109/ICASSP43922.2022.9747231.

[11] H. Beigi. *Fundamentals of Speaker Recognition*. Springer US, 2011. DOI: 10.1007/978-0-387-77592-0. URL: http://dx.doi.org/10.1007/978-0-387-77592-0.

[12] J. H. Hansen. "Analysis and compensation of speech under stress and noise for environmental robustness in speech recognition". In: *Speech Communication* 20.1 (1996). Speech under Stress, pp. 151–173. ISSN: 0167-6393. DOI: https://doi.org/10.1016/S0167-6393(96)00050-7. URL: https://www.sciencedirect.com/science/article/pii/S0167639396000507.

[13] A. Ikeno, V. Varadarajan, S. Patil, and J. H. Hansen. "UT-Scope: Speech under Lombard Effect and Cognitive Stress". In: *2007 IEEE Aerospace Conference*. 2007, pp. 1–7. DOI: 10.1109/AERO.2007.352975.

[14] F. Kelly and J. H. Hansen. "Analysis and Calibration of Lombard Effect and Whisper for Speaker Recognition". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 29 (2021), pp. 927–942. DOI: 10.1109/TASLP.2021.3053388.

[15] R. Pappagari, T. Wang, J. Villalba, N. Chen, and N. Dehak. "X-Vectors Meet Emotions: A Study On Dependencies Between Emotion and Speaker Recognition". In: *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2020, pp. 7169–7173. DOI: 10.1109/ICASSP40776.2020.9054317.

[16] J. H. Hansen and T. Hasan. "Speaker Recognition by Machines and Humans: A tutorial review". In: *IEEE Signal Processing Magazine* 32.6 (2015), pp. 74–99. DOI: 10.1109/MSP.2015.2462851.

[17] M. FarrÚs. “Voice Disguise in Automatic Speaker Recognition”. In: *ACM Comput. Surv.* 51.4 (July 2018). ISSN: 0360-0300. DOI: 10.1145/3195832. URL: https://doi.org/10.1145/3195832.

[18] L. F. Gallardo, M. Wagner, and S. Möller. “Transmission channel effects on human speaker identification in multiparty conference calls”. In: *2013 8th International Conference on Information Technology in Asia (CITA)*. 2013, pp. 1–6. DOI: 10.1109/CITA.2013.6637569.

[19] A. Gusev, A. Vinogradova, S. Novoselov, and S. Astapov. “SdSVC Challenge 2021: Tips and Tricks to Boost the Short-Duration Speaker Verification System Performance”. In: *Proc. Interspeech 2021*. 2021, pp. 2307–2311. DOI: 10.21437/Interspeech.2021-1737.

[20] Y. Zheng, J. Peng, Y. Chen, Y. Zhang, J. Wang, M. Liu, and M. Xu. “The SpeakIn Speaker Verification System for Far-Field Speaker Verification Challenge 2022”. In: *Proc. The 2022 Far-field Speaker Verification Challenge (FFSVC2022)*. 2022, pp. 15–19. DOI: 10.21437/FFSVC.2022-4.

[21] A. Amani, M. Mohammadamini, and H. Veisi. “Kurdish Spoken Dialect Recognition Using X-Vector Speaker Embedding”. In: *Speech and Computer*. Ed. by A. Karpov and R. Potapova. Cham: Springer International Publishing, 2021, pp. 50–57.

[22] V. Brignatz, J. Duret, D. Matrouf, and M. Rouvier. “Language Adaptation for Speaker Recognition Systems Using Contrastive Learning”. In: *Speech and Computer*. Ed. by A. Karpov and R. Potapova. Cham: Springer International Publishing, 2021, pp. 91–99.

[23] D. Raj, D. Snyder, D. Povey, and S. Khudanpur. “Probing the Information Encoded in X-Vectors”. In: *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. 2019, pp. 726–733. DOI: 10.1109/ASRU46091.2019.9003979.

[24] J. H. Hansen. “Analysis and compensation of speech under stress and noise for environmental robustness in speech recognition”. In: *Speech Communication* 20.1 (1996). Speech under Stress, pp. 151–173. ISSN: 0167-6393. DOI: https://doi.org/10.1016/S0167-6393(96)00050-7. URL: https://www.sciencedirect.com/science/article/pii/S0167639396000507.

[25] “Effect of sound absorption and screen height on spatial decay of speech Experimental study in an open-plan office”. In: *Applied Acoustics* 166 (2020), p. 107340. ISSN: 0003-682X. DOI: https://doi.org/10.1016/j.apacoust.2020.107340.

[26] P. C. Loizou. *Speech Enhancement: Theory and Practice*. Ed. by S. E. (2nd. CRC Press, 2013.

[27] M.-W. Mak and H.-B. Yu. “A study of voice activity detection techniques for NIST speaker recognition evaluations”. In: *Computer Speech Language* 28.1 (2014), pp. 295–313. ISSN: 0885-2308. DOI: https://doi.org/10.1016/j.csl.2013.07.003. URL: https://www.sciencedirect.com/science/article/pii/S0885230813000533.

[28] S. Ding, Q. Wang, S.-Y. Chang, L. Wan, and I. Lopez Moreno. “Personal VAD: Speaker-Conditioned Voice Activity Detection”. In: *Proc. The Speaker and Language Recognition Workshop (Odyssey 2020)*. 2020, pp. 433–439. DOI: 10.21437/Odyssey.2020-62.

[29] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely. “The Kaldi Speech Recognition Toolkit”. In: *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Catalog No.: CFP11SRW-USB. Hilton Waikoloa Village, Big Island, Hawaii, US: IEEE Signal Processing Society, Dec. 2011.

[30] L.-C. Chang and J.-W. Hung. “A Preliminary Study of Robust Speech Feature Extraction Based on Maximizing the Probability of States in Deep Acoustic Models”. In: *Applied System Innovation* 5.4 (2022). ISSN: 2571-5577. URL: https://www.mdpi.com/2571-5577/5/4/71.

[31] X. Huang, A. Acero, H.-W. Hon, and R. Reddy. *Spoken Language Processing: A Guide to Theory, Algorithm, and System Development*. 1st. USA: Prentice Hall PTR, 2001. ISBN: 0130226165.

[32] Z. Bai and X.-L. Zhang. “Speaker recognition based on deep learning: An overview”. In: *Neural Networks* 140 (2021), pp. 65–99. ISSN: 0893-6080. DOI: https://doi.org/10.1016/j.neunet.2021.03.004. URL: https://www.sciencedirect.com/science/article/pii/S0893608021000848.

[33] S. Schneider, A. Baevski, R. Collobert, and M. Auli. “wav2vec: Unsupervised Pre-Training for Speech Recognition”. In: *Proc. Interspeech 2019*. 2019, pp. 3465–3469. DOI: 10.21437/Interspeech.2019-1873.

[34] Z. Fan, M. Li, S. Zhou, and B. Xu. “Exploring wav2vec 2.0 on Speaker Verification and Language Identification”. In: *Proc. Interspeech 2021*. 2021, pp. 1509–1513. DOI: 10.21437/Interspeech.2021-1280.

[35] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli. “wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations”. In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin. Vol. 33. Curran Associates, Inc., 2020,

pp. 12449–12460. URL: https://proceedings.neurips.cc/paper/2020/file/92d1e1eb1cd6f9fba3227870bb6d7f07-Paper.pdf.

[36] S. Chen, C. Wang, Z. Chen, Y. Wu, S. Liu, Z. Chen, J. Li, N. Kanda, T. Yoshioka, X. Xiao, J. Wu, L. Zhou, S. Ren, Y. Qian, Y. Qian, J. Wu, M. Zeng, X. Yu, and F. Wei. "WavLM: Large-Scale Self-Supervised Pre-Training for Full Stack Speech Processing". In: *IEEE Journal of Selected Topics in Signal Processing* 16.6 (2022), pp. 1505–1518. DOI: 10.1109/JSTSP.2022.3188113.

[37] P. Kenny, G. Boulianne, P. Ouellet, and P. Dumouchel. "Joint Factor Analysis Versus Eigenchannels in Speaker Recognition". In: *IEEE Transactions on Audio, Speech, and Language Processing* 15.4 (2007), pp. 1435–1447. DOI: 10.1109/TASL.2006.881693.

[38] W. Campbell, D. Sturim, D. Reynolds, and A. Solomonoff. "SVM Based Speaker Verification using a GMM Supervector Kernel and NAP Variability Compensation". In: *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*. Vol. 1. 2006, pp. I–I. DOI: 10.1109/ICASSP.2006.1659966.

[39] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet. "Front-End Factor Analysis for Speaker Verification". In: *IEEE Transactions on Audio, Speech, and Language Processing* 19.4 (2011), pp. 788–798. DOI: 10.1109/TASL.2010.2064307.

[40] S. Yaman, J. Pelecanos, and R. Sarikaya. "Bottleneck features for speaker recognition". In: *Proc. The Speaker and Language Recognition Workshop (Odyssey 2012)*. 2012, pp. 105–108.

[41] A. Lozano-Diez, A. Silnova, P. Matejka, O. Glembek, O. Plchot, J. Pesan, L. Burget, and J. Gonzalez-Rodriguez. "Analysis and Optimization of Bottleneck Features for Speaker Recognition". In: pp. 352–357.

[42] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. Lang. "Phoneme recognition using time-delay neural networks". In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 37.3 (1989), pp. 328–339. DOI: 10.1109/29.21701.

[43] M.-W. Mak and J.-T. Chien. "Deep Learning Models". In: *Machine Learning for Speaker Recognition*. Cambridge University Press, 2020, pp. 115–168. DOI: 10.1017/9781108552332.005.

[44] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[45] K. Simonyan and A. Zisserman. "Very deep convolutional networks for large-scale image recognition". In: *arXiv preprint arXiv:1409.1556* (2014).

[46] K. He, X. Zhang, S. Ren, and J. Sun. "Deep Residual Learning for Image Recognition". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778. DOI: 10.1109/CVPR.2016.90.

[47] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, and R. Pang. "Conformer: Convolution-augmented Transformer for Speech Recognition". In: *Proc. Interspeech 2020*. 2020, pp. 5036–5040. DOI: 10.21437/Interspeech.2020-3015.

[48] Z. Huang, S. Wang, and K. Yu. "Angular Softmax for Short-Duration Text-independent Speaker Verification". In: *Proc. Interspeech 2018*. 2018, pp. 3623–3627. DOI: 10.21437/Interspeech.2018-1545.

[49] Y.-Q. Yu, L. Fan, and W.-J. Li. "Ensemble Additive Margin Softmax for Speaker Verification". In: *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2019, pp. 6046–6050. DOI: 10.1109/ICASSP.2019.8683649.

[50] X. Xiang, S. Wang, H. Huang, Y. Qian, and K. Yu. "Margin Matters: Towards More Discriminative Deep Neural Network Embeddings for Speaker Recognition". In: *2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. 2019, pp. 1652–1656. DOI: 10.1109/APSIPAASC47483.2019.9023039.

[51] "Linear discriminant analysis-a brief tutorial". In: Institute for Signal and information Processing, 1998/3/2.

[52] A. Tharwat, T. Gaber, A. Ibrahim, and A. E. Hassanien. "Linear discriminant analysis: A detailed tutorial". In: *AI Communications* 30.2 (May 2017), pp. 169–190. DOI: 10.3233/aic-170729. URL: https://doi.org/10.3233/aic-170729.

[53] M. Mohammad Amini and D. Matrouf. "Data augmentation versus noise compensation for x-vector speaker recognition systems in noisy environments". In: *2020 28th European Signal Processing Conference (EUSIPCO)*. 2021, pp. 1–5. DOI: 10.23919/Eusipco47968.2020.9287690.

[54] M. MohammadAmini, D. Matrouf, J.-F. Bonatsre, S. Dowerah, R. Serizel, and D. Jouvet. "A Comprehensive Exploration of Noise Robustness and Noise Compensation in ResNet and TDNN-based Speaker Recognition Systems". In: *2022 30th European Signal Processing Conference (EUSIPCO)*. 2022, pp. 364–368. DOI: 10.23919/EUSIPCO55093.2022.9909726.

[55] A. Martin, G. Doddington, T. Kamm, M. Ordowski, and M. Przybocki. "The DET curve in assessment of detection task performance". In: *Proc. 5th European Conference on Speech Communication and Technology (Eurospeech 1997)*. 1997, pp. 1895–1898. DOI: 10.21437/Eurospeech.1997-504.

[56] O. Sadjadi, C. Greenberg, E. Singer, L. Mason, and D. Reynolds. *NIST 2021 Speaker Recognition Evaluation Plan*. en. 2021-07-12 04:07:00 2021. URL: https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=932697.

[57] T. Ko, V. Peddinti, D. Povey, M. L. Seltzer, and S. Khudanpur. "A study on data augmentation of reverberant speech for robust speech recognition". In: *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2017, pp. 5220–5224. DOI: 10.1109/ICASSP.2017.7953152.

[58] M. Mclaren, D. Castán, M. K. Nandwana, L. Ferrer, and E. Yilmaz. "How to train your speaker embeddings extractor ". In: *Proc. The Speaker and Language Recognition Workshop (Odyssey 2018)*. 2018, pp. 327–334. DOI: 10.21437/Odyssey.2018-46.

[59] O. Novotný, O. Plchot, P. Matjka, L. Moner, and O. Glembek. "On the use of X-vectors for Robust Speaker Recognition". In: *Proc. Odyssey 2018 The Speaker and Language Recognition Workshop*. 2018, pp. 168–175. DOI: 10.21437/Odyssey.2018-24. URL: http://dx.doi.org/10.21437/Odyssey.2018-24.

[60] A. Kanagasundaram, S. Sridharan, G. Sriram, S. Prachi, and C. Fookes. "A Study of x-Vector Based Speaker Recognition on Short Utterances". In: *Proc. Interspeech 2019*. 2019, pp. 2943–2947. DOI: 10.21437/Interspeech.2019-1891.

[61] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le. "SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition". In: *Proc. Interspeech 2019*. 2019, pp. 2613–2617. DOI: 10.21437/Interspeech.2019-2680.

[62] S. Wang, J. Rohdin, O. Plchot, L. Burget, K. Yu, and J. ernocký. "Investigation of Specaugment for Deep Speaker Embedding Learning". In: *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2020, pp. 7139–7143. DOI: 10.1109/ICASSP40776.2020.9053481.

[63] E. Kharitonov, M. Rivière, G. Synnaeve, L. Wolf, P.-E. Mazaré, M. Douze, and E. Dupoux. "Data Augmenting Contrastive Learning of Speech Representations in the Time Domain". In: *2021 IEEE Spoken Language Technology Workshop (SLT)*. 2021, pp. 215–222. DOI: 10.1109/SLT48900.2021.9383605.

[64] A. van den Oord, Y. Li, and O. Vinyals. "Representation Learning with Contrastive Predictive Coding". In: *ArXiv* abs/1807.03748 (2018).

[65] H. Tak, M. Kamble, J. Patino, M. Todisco, and N. Evans. "Rawboost: A Raw Data Boosting and Augmentation Method Applied to Automatic Speaker Verification Anti-Spoofing". In: *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2022, pp. 6382–6386. DOI: 10.1109/ICASSP43922.2022.9746213.

[66] H. Tak, M. Todisco, X. Wang, J.-w. Jung, J. Yamagishi, and N. Evans. "Automatic Speaker Verification Spoofing and Deepfake Detection Using Wav2vec 2.0 and Data Augmentation". In: *Proc. The Speaker and Language Recognition Workshop (Odyssey 2022)*. 2022, pp. 112–119. DOI: 10.21437/Odyssey.2022-16.

[67] W. Lin and M.-W. Mak. "Robust Speaker Verification Using Population-Based Data Augmentation". In: *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2022, pp. 7642–7646. DOI: 10.1109/ICASSP43922.2022.9746956.

[68] D. Wang and J. Chen. "Supervised Speech Separation Based on Deep Learning: An Overview". In: *IEEE/ACM transactions on audio, speech, and language processing* 26.10 (Oct. 2018), pp. 1702–1726. ISSN: 2329-9290. DOI: 10.1109/taslp.2018.2842159. URL: https://europepmc.org/articles/PMC6586438.

[69] E. Vincent, R. Gribonval, and C. Fevotte. "Performance measurement in blind audio source separation". In: *IEEE Transactions on Audio, Speech, and Language Processing* 14.4 (2006), pp. 1462–1469. DOI: 10.1109/TSA.2005.858005.

[70] D. Wang and J. Chen. "Supervised Speech Separation Based on Deep Learning: An Overview". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 26.10 (2018), pp. 1702–1726. DOI: 10.1109/TASLP.2018.2842159.

[71] N. Furnon, R. Serizel, S. Essid, and I. Illina. "DNN-Based Mask Estimation for Distributed Speech Enhancement in Spatially Unconstrained Microphone Arrays". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 29 (2021), pp. 2310–2323. DOI: 10.1109/TASLP.2021.3092838.

[72] S. Shon, H. Tang, and J. Glass. "VoiceID Loss: Speech Enhancement for Speaker Verification". In: *Proc. Interspeech 2019*. 2019, pp. 2888–2892. DOI: 10.21437/Interspeech.2019-1496.

[73] Q. Wang, K. A. Lee, T. Koshinaka, K. Okabe, and H. Yamamoto. "Task-aware Warping Factors in Mask-based Speech Enhancement". In: *2021 29th European Signal Processing Conference (EUSIPCO)*. 2021, pp. 476–480. DOI: 10.23919/EUSIPCO54536.2021.9616081.

[74] S. E. Eskimez, P. Soufleris, Z. Duan, and W. Heinzelman. “Front-end speech enhancement for commercial speaker verification systems”. In: *Speech Communication* 99 (2018), pp. 101–113. ISSN: 0167-6393. DOI: https://doi.org/10.1016/j.specom.2018.03.008. URL: https://www.sciencedirect.com/science/article/pii/S0167639317302480.

[75] H. Taherian, Z.-Q. Wang, J. Chang, and D. Wang. “Robust Speaker Recognition Based on Single-Channel and Multi-Channel Speech Enhancement”. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 28 (2020), pp. 1293–1302. DOI: 10.1109/TASLP.2020.2986896.

[76] J. Chang and D. Wang. “Robust speaker recognition based on DNN/i-vectors and speech separation”. In: *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2017, pp. 5415–5419. DOI: 10.1109/ICASSP.2017.7953191.

[77] H. Taherian, Z.-Q. Wang, and D. Wang. “Deep Learning Based Multi-Channel Speaker Recognition in Noisy and Reverberant Environments”. In: *Proc. Interspeech 2019*. 2019, pp. 4070–4074. DOI: 10.21437/Interspeech.2019-1428.

[78] Z. Gao, M. Mak, and W. Lin. “UNet-DenseNet for Robust Far-Field Speaker Verification”. In: *Proc. Interspeech 2022*. 2022, pp. 3714–3718. DOI: 10.21437/Interspeech.2022-10350.

[79] S. Dowerah, R. Serizel, D. Jouvet, M. Mohammadamini, and D. Matrouf. “Joint Optimization of Diffusion Probabilistic-Based Multichannel Speech Enhancement with Far-Field Speaker Verification”. In: *2022 IEEE Spoken Language Technology Workshop (SLT)*. 2023, pp. 428–435. DOI: 10.1109/SLT54892.2023.10022350.

[80] F. Zhao, H. Li, and X. Zhang. “A Robust Text-independent Speaker Verification Method Based on Speech Separation and Deep Speaker”. In: *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2019, pp. 6101–6105. DOI: 10.1109/ICASSP.2019.8683762.

[81] O. Plchot, L. Burget, H. Aronowitz, and P. Matëjka. “Audio enhancing with DNN autoencoder for speaker recognition”. In: *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2016, pp. 5090–5094. DOI: 10.1109/ICASSP.2016.7472647.

[82] O. Novotný, O. Plchot, O. Glembek, J. . ernocký, and L. Burget. “Analysis of DNN Speech Signal Enhancement for Robust Speaker Recognition”. In: *Computer Speech Language* 58 (2019), pp. 403–421. ISSN: 0885-2308.

DOI: https://doi.org/10.1016/j.csl.2019.06.004. URL: https://www.sciencedirect.com/science/article/pii/S0885230818303607.

[83] Z. Oo, Y. Kawakami, L. Wang, S. Nakagawa, X. Xiao, and M. Iwahashi. "DNN-Based Amplitude and Phase Feature Enhancement for Noise Robust Speaker Identification". In: *Proc. Interspeech 2016*. 2016, pp. 2204–2208. DOI: 10.21437/Interspeech.2016-717.

[84] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. "Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems*. Ed. by Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger. Vol. 27. Curran Associates, Inc., 2014. URL: https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf.

[85] D. Michelsanti and Z.-H. Tan. "Conditional Generative Adversarial Networks for Speech Enhancement and Noise-Robust Speaker Verification". In: *Proc. Interspeech 2017*. 2017, pp. 2008–2012. DOI: 10.21437/Interspeech.2017-1620.

[86] H. Yu, T. Hu, Z. Ma, Z.-H. Tan, and J. Guo. "Multi-Task Adversarial Network Bottleneck Features for Noise-Robust Speaker Verification". In: *2018 International Conference on Network Infrastructure and Digital Content (IC-NIDC)*. 2018, pp. 165–169. DOI: 10.1109/ICNIDC.2018.8525526.

[87] J. Zhou, T. Jiang, L. Li, Q. Hong, Z. Wang, and B. Xia. "Training Multi-task Adversarial Network for Extracting Noise-robust Speaker Embedding". In: *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2019, pp. 6196–6200. DOI: 10.1109/ICASSP.2019.8683828.

[88] J. Zhou, T. Jiang, Q. Hong, and L. Li. "Extraction of Noise-Robust Speaker Embedding Based on Generative Adversarial Networks". In: *2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. 2019, pp. 1641–1645. DOI: 10.1109/APSIPAASC47483.2019.9023295.

[89] X. Fang, L. Zou, J. Li, L. Sun, and Z.-H. Ling. "Channel Adversarial Training for Cross-channel Text-independent Speaker Recognition". In: *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2019, pp. 6221–6225. DOI: 10.1109/ICASSP.2019.8682327.

[90] Z. Chen, S. Wang, Y. Qian, and K. Yu. "Channel Invariant Speaker Embedding Learning with Joint Multi-Task and Adversarial Training". In: *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2020, pp. 6574–6578. DOI: 10.1109/ICASSP40776.2020.9053905.

[91] C. Luu, P. Bell, and S. Renals. "Channel Adversarial Training for Speaker Verification and Diarization". In: *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2020, pp. 7094–7098. DOI: 10.1109/ICASSP40776.2020.9053323.

[92] G. Bhattacharya, J. Alam, and P. Kenny. "Adapting End-to-end Neural Speaker Verification to New Languages and Recording Conditions with Adversarial Training". In: *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2019, pp. 6041–6045. DOI: 10.1109/ICASSP.2019.8682611.

[93] S. Wang, J. Rohdin, L. Burget, O. Plchot, Y. Qian, K. Yu, and J. ernocký. "On the Usage of Phonetic Information for Text-Independent Speaker Embedding Extraction". In: *Proc. Interspeech 2019*. 2019, pp. 1148–1152. DOI: 10.21437/Interspeech.2019-3036.

[94] B. Sun and K. Saenko. "Deep CORAL: Correlation Alignment for Deep Domain Adaptation". In: *Computer Vision – ECCV 2016 Workshops*. Ed. by G. Hua and H. Jégou. Cham: Springer International Publishing, 2016, pp. 443–450.

[95] R. Li, W. Zhang, and D. Chen. "The Coral++ Algorithm for Unsupervised Domain Adaptation of Speaker Recognition". In: *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2022, pp. 7172–7176. DOI: 10.1109/ICASSP43922.2022.9747792.

[96] W. Lin, M.-M. Mak, N. Li, D. Su, and D. Yu. "Multi-Level Deep Neural Network Adaptation for Speaker Verification Using MMD and Consistency Regularization". In: *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2020, pp. 6839–6843. DOI: 10.1109/ICASSP40776.2020.9054134.

[97] R. Duroselle, D. Jouvet, and I. Illina. "Unsupervised Regularization of the Embedding Extractor for Robust Language Identification". In: *Proc. The Speaker and Language Recognition Workshop (Odyssey 2020)*. 2020, pp. 39–46. DOI: 10.21437/Odyssey.2020-6.

[98] S. Wang, Y. Yang, Y. Qian, and K. Yu. "Revisiting the Statistics Pooling Layer in Deep Speaker Embedding Learning". In: *2021 12th International Symposium on Chinese Spoken Language Processing (ISCSLP)*. 2021, pp. 1–5. DOI: 10.1109/ISCSLP49672.2021.9362097.

[99] A. Silnova, T. Stafylakis, L. Moner, O. Plchot, J. Rohdin, P. Matjka, L. Burget, O. Glembek, and N. Brummer. "Analyzing Speaker Verification Embedding Extractors and Back-Ends Under Language and Channel Mismatch". In: *Proc. The Speaker and Language Recognition Workshop (Odyssey 2022)*. 2022, pp. 9–16. DOI: 10.21437/Odyssey.2022-2.

[100] Y. Zhu, T. Ko, D. Snyder, B. Mak, and D. Povey. "Self-Attentive Speaker Embeddings for Text-Independent Speaker Verification". In: *Proc. Interspeech 2018*. 2018, pp. 3573–3577. DOI: 10.21437/Interspeech.2018-1158.

[101] M. Sang, Y. Zhao, G. Liu, J. H. L. Hansen, and J. Wu. *Improving Transformer-based Networks With Locality For Automatic Speaker Verification*. 2023. DOI: 10.48550/ARXIV.2302.08639. URL: https://arxiv.org/abs/2302.08639.

[102] Y. Shi, Q. Huang, and T. Hain. "H-VECTORS: Improving the robustness in utterance-level speaker embeddings using a hierarchical attention model". In: *Neural Networks* 142 (2021), pp. 329–339. ISSN: 0893-6080. DOI: https://doi.org/10.1016/j.neunet.2021.05.024. URL: https://www.sciencedirect.com/science/article/pii/S0893608021002203.

[103] Y. Shi, Q. Huang, and T. Hain. "Weakly Supervised Training of Hierarchical Attention Networks for Speaker Identification". In: *Proc. Interspeech 2020*. 2020, pp. 2992–2996. DOI: 10.21437/Interspeech.2020-1774.

[104] Y. Shi, Q. Huang, and T. Hain. "Robust Speaker Recognition Using Speech Enhancement And Attention Model". In: *Proc. The Speaker and Language Recognition Workshop (Odyssey 2020)*. 2020, pp. 451–458. DOI: 10.21437/Odyssey.2020-65.

[105] W. Ben Kheder, D. Matrouf, P.-M. Bousquet, J.-F. Bonastre, and M. Ajili. "Fast i-vector denoising using MAP estimation and a noise distributions database for robust speaker recognition". In: *Computer Speech Language* 45 (2017), pp. 104–122. ISSN: 0885-2308. DOI: https://doi.org/10.1016/j.csl.2016.12.007. URL: https://www.sciencedirect.com/science/article/pii/S088523081730030X.

[106] W. B. Kheder, D. Matrouf, M. Ajili, and J.-F. Bonastre. "A Unified Joint Model to Deal With Nuisance Variabilities in the i-Vector Space". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 26.3 (2018), pp. 633–645. DOI: 10.1109/TASLP.2018.2789399.

[107] W. B. Kheder, D. Matrouf, M. Ajili, and J.-F. Bonastre. "Probabilistic Approach Using Joint Clean and Noisy i-Vectors Modeling for Speaker Recognition". In: *Proc. Interspeech 2016*. 2016, pp. 3638–3642. DOI: 10.21437/Interspeech.2016-1292.

[108] M. J. Alam, G. Bhattacharya, and P. Kenny. “Speaker Verification in Mismatched Conditions with Frustratingly Easy Domain Adaptation ”. In: *Proc. The Speaker and Language Recognition Workshop (Odyssey 2018)*. 2018, pp. 176–180. DOI: 10.21437/Odyssey.2018-25.

[109] T. Pekhovsky, S. Novoselov, A. Sholohov, and O. Kudashev. “On autoencoders in the i-vector space for speaker recognition”. In: *Proc. The Speaker and Language Recognition Workshop (Odyssey 2016)*. 2016, pp. 217–224. DOI: 10.21437/Odyssey.2016-31.

[110] S. Mahto, H. Yamamoto, and T. Koshinaka. “i-Vector Transformation Using a Novel Discriminative Denoising Autoencoder for Noise-Robust Speaker Recognition”. In: *Proc. Interspeech 2017*. 2017, pp. 3722–3726. DOI: 10.21437/Interspeech.2017-731.

[111] R. Font. “A Denoising Autoencoder for Speaker Recognition. Results on the MCE 2018 Challenge”. In: *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2019, pp. 6016–6020. DOI: 10.1109/ICASSP.2019.8683525.

[112] S. Rozenberg, H. Aronowitz, and R. Hoory. “Siamese X-Vector Reconstruction for Domain Adapted Speaker Recognition”. In: *Proc. Interspeech 2020*. 2020, pp. 1526–1529. DOI: 10.21437/Interspeech.2020-1742.

[113] W. Lin, M.-W. Mak, L. Li, and J.-T. Chien. “Reducing Domain Mismatch by Maximum Mean Discrepancy Based Autoencoders ”. In: *Proc. Odyssey 2018 The Speaker and Language Recognition Workshop*. 2018, pp. 162–167. DOI: 10.21437/Odyssey.2018-23. URL: http://dx.doi.org/10.21437/Odyssey.2018-23.

[114] W.-w. Lin, M.-W. Mak, and J.-T. Chien. “Multisource I-Vectors Domain Adaptation Using Maximum Mean Discrepancy Based Autoencoders”. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 26.12 (2018), pp. 2412–2422. DOI: 10.1109/TASLP.2018.2866707.

[115] Q. Wang, W. Rao, S. Sun, L. Xie, E. S. Chng, and H. Li. “Unsupervised Domain Adaptation via Domain Adversarial Training for Speaker Recognition”. In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2018, pp. 4889–4893. DOI: 10.1109/ICASSP.2018.8461423.

[116] Y. Tu, M.-W. Mak, and J.-T. Chien. “Variational Domain Adversarial Learning for Speaker Verification”. In: *Proc. Interspeech 2019*. 2019, pp. 4315–4319. DOI: 10.21437/Interspeech.2019-2168.

[117] P. S. Nidadavolu, J. Villalba, and N. Dehak. "Cycle-GANs for Domain Adaptation of Acoustic Features for Speaker Recognition". In: *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2019, pp. 6206–6210. DOI: 10.1109/ICASSP.2019.8683055.

[118] P. S. Nidadavolu, S. Kataria, J. Villalba, and N. Dehak. "Low-Resource Domain Adaptation for Speaker Recognition Using Cycle-Gans". In: *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. 2019, pp. 710–717. DOI: 10.1109/ASRU46091.2019.9003748.

[119] P. S. Nidadavolu, S. Kataria, J. Villalba, and N. Dehak. "Low-Resource Domain Adaptation for Speaker Recognition Using Cycle-Gans". In: *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. 2019, pp. 710–717. DOI: 10.1109/ASRU46091.2019.9003748.

[120] A. Nagrani, J. S. Chung, W. Xie, and A. Zisserman. "Voxceleb: Large-scale speaker verification in the wild". In: *Computer Speech Language* 60 (2020), p. 101027. ISSN: 0885-2308. DOI: https://doi.org/10.1016/j.csl.2019.101027. URL: https://www.sciencedirect.com/science/article/pii/S0885230819302712.

[121] D. Snyder, G. Chen, and D. Povey. *MUSAN: A Music, Speech, and Noise Corpus*. arXiv:1510.08484v1. 2015. eprint: 1510.08484.

[122] E. Fonseca, J. Pons, X. Favory, F. Font, D. Bogdanov, A. Ferraro, S. Oramas, A. Porter, and X. Serra. "Freesound Datasets: A Platform for the Creation of Open Audio Datasets". In: *International Society for Music Information Retrieval Conference*. 2017.

[123] O. Kudashev, S. Novoselov, K. Simonchik, and A. Kozlov. "A Speaker Recognition System for the SITW Challenge". In: *Proc. Interspeech 2016*. 2016, pp. 833–837. DOI: 10.21437/Interspeech.2016-1197.

[124] D. Garcia-Romero, D. Snyder, S. Watanabe, G. Sell, A. McCree, D. Povey, and S. Khudanpur. "Speaker Recognition Benchmark Using the CHiME-5 Corpus". In: *Proc. Interspeech 2019*. 2019, pp. 1506–1510. DOI: 10.21437/Interspeech.2019-2174.

[125] M. Ajili, J.-F. Bonastre, J. Kahn, S. Rossato, and G. Bernard. "FABIOLE, a Speech Database for Forensic Speaker Comparison". In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*. Portoro, Slovenia: European Language Resources Association (ELRA), May 2016, pp. 726–733. URL: https://aclanthology.org/L16-1115.

[126] M. Mohammadamini, D. Matrouf, and P.-G. Noé. "Denoising x-vectors for Robust Speaker Recognition". In: *Proc. The Speaker and Language Recognition Workshop (Odyssey 2020)*. 2020, pp. 75–80. DOI: 10.21437/Odyssey.2020-11.

[127] M. Mohammadamini, D. Matrouf, J.-F. Bonastre, R. Serizel, S. Dowerah, and D. Jouvet. "Compensate multiple distortions for speaker recognition systems". In: *2021 29th European Signal Processing Conference (EUSIPCO)*. 2021, pp. 141–145. DOI: 10.23919/EUSIPCO54536.2021.9615983.

[128] L. Chai, J. Du, and Y.-n. Wang. "Gaussian density guided deep neural network for single-channel speech enhancement". In: *2017 IEEE 27th International Workshop on Machine Learning for Signal Processing (MLSP)*. 2017, pp. 1–6. DOI: 10.1109/MLSP.2017.8168116.

[129] K. Zhang, W. Zuo, S. Gu, and L. Zhang. "Learning Deep CNN Denoiser Prior for Image Restoration". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 2808–2817. DOI: 10.1109/CVPR.2017.300.

[130] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Y. Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: https://www.tensorflow.org/.

[131] F. Chollet *et al. Keras*. https://keras.io. 2015.

[132] M. Mohammadamini, D. Matrouf, S. Dowerah, R. Serizel, D. Jouvet, and J.-F. Bonastre. "Le comportement des systèmes de reconnaissance du locuteur de l'état de l'art face aux variabilités acoustiques". In: *Proc. XXXIVe Journées d'Études sur la Parole – JEP 2022*. 2022, pp. 241–249. DOI: 10.21437/JEP.2022-26.

[133] M. MohammadAmini, D. Matrouf, J.-F. Bonastre, S. Dowerah, R. Serizel, and D. Jouvet. "Learning Noise Robust ResNet-Based Speaker Embedding for Speaker Recognition". In: *Proc. The Speaker and Language Recognition Workshop (Odyssey 2022)*. 2022, pp. 41–46. DOI: 10.21437/Odyssey.2022-6.

[134] M. Mohammadamini, D. Matrouf, J.-F. Bonastre, S. Dowerah, R. Serizel, and D. Jouvet. "Barlow Twins self-supervised learning for robust speaker recognition". In: *Proc. Interspeech 2022*. 2022, pp. 4033–4037. DOI: 10.21437/Interspeech.2022-11301.

[135] Y. Ma, K. A. Lee, V. Hautamäki, and H. Li. "PL-EESR: PERCEPTUAL LOSS BASED END-TO-END ROBUST SPEAKER REPRESENTATION EXTRACTION". In: *ASRU*. Sept. 2021.

[136] Q. Wang, K. Okabe, K. A. Lee, and T. Koshinaka. "A Generalized Framework for Domain Adaptation of PLDA in Speaker Recognition". In: *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2020, pp. 6619–6623. DOI: 10.1109/ICASSP40776.2020.9054113.

[137] J. Zbontar, L. Jing, I. Misra, Y. LeCun, and S. Deny. "Barlow Twins: Self-Supervised Learning via Redundancy Reduction". In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by M. Meila and T. Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, 18–24 Jul 2021, pp. 12310–12320. URL: https://proceedings.mlr.press/v139/zbontar21a.html.

[138] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. "A Simple Framework for Contrastive Learning of Visual Representations". In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by H. D. III and A. Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, 13–18 Jul 2020, pp. 1597–1607. URL: https://proceedings.mlr.press/v119/chen20j.html.

[139] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, E. Buchatskaya, C. Doersch, B. A. Pires, Z. D. Guo, M. G. Azar, B. Piot, K. Kavukcuoglu, R. Munos, and M. Valko. *Bootstrap your own latent: A new approach to self-supervised Learning*. 2020. DOI: 10.48550/ARXIV.2006.07733. URL: https://arxiv.org/abs/2006.07733.

[140] C. Wu, F. Wu, and Y. Huang. *Rethinking InfoNCE: How Many Negative Samples Do You Need?* 2021. DOI: 10.48550/ARXIV.2105.13003. URL: https://arxiv.org/abs/2105.13003.

[141] Y.-H. H. Tsai, S. Bai, L.-P. Morency, and R. Salakhutdinov. *A Note on Connecting Barlow Twins with Negative-Sample-Free Contrastive Learning*. 2021. DOI: 10.48550/ARXIV.2104.13712. URL: https://arxiv.org/abs/2104.13712.

[142] J. Deng, J. Guo, N. Xue, and S. Zafeiriou. "ArcFace: Additive Angular Margin Loss for Deep Face Recognition". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 4685–4694. DOI: 10.1109/CVPR.2019.00482.